
Memetische evolutionäre Optimierung von Hochwasserspeichersystemen

Vom Fachbereich Bauingenieurwesen und Geodäsie
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.) genehmigte Dissertation



TECHNISCHE
UNIVERSITÄT
DARMSTADT

vorgelegt von
Dipl.-Ing. Christoph Hübner
aus Groß-Umstadt

Referent: Prof. Dr.-Ing. Manfred Ostrowski
Korreferent: Prof. Dr. rer. nat. Robert Jüpner

Eingereicht am: 22.01.2010
Mündliche Prüfung am: 18.05.2010

Darmstadt, 2010

D 17

Memetic Evolutionary Optimization of Multi Reservoir Systems

Christoph Huebner



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Kurzfassung

Die Planung von Hochwasserspeichern im Hinblick auf Lage, Volumina und Steuerung ist eine komplexe ingenieurtechnische Aufgabe, da die Interaktion von Maßnahmen wie Hochwasserrückhaltebecken, Dämme oder Renaturierungsmaßnahmen und ihre Wirkung auf den Abflussprozess zu berücksichtigen ist. In dicht besiedelten Gebieten besteht meist keine große Freiheit in Planung und Entwurf. Daher müssen Hochwassermanagement-Projekte in einen umfassenden Planungsprozess eingebettet werden. Hochwassermanagement betrifft nicht nur technisch-ökonomische Bereiche der Gesellschaft, sondern auch das jeweils natürliche und soziale Umfeld. Zusätzlich kann der Klimawandel Veränderungen des Niederschlagsregimes mit sich bringen, welche entscheidenden Einfluss auf Hochwasserentstehung und -ablauf haben können. Äußere Einflüsse dieser Art sind kaum prognostizierbar, jedoch müssen sie vorsorglich in vorausschauende Planungsprozesse eingebunden werden. Insgesamt sind Entscheidungsprozesse im Hochwassermanagement erheblich komplexer und interdisziplinärer geworden – nicht zuletzt auch durch die Einbeziehung der Öffentlichkeit. Es ist dringend erforderlich, Methoden und Algorithmen bereitzustellen, die diese Entscheidungsprozesse beschleunigen und transparent gestalten.

Eine wichtige Herausforderung stellen die neuen Richtlinien der Europäischen Gemeinschaft dar. Die auf die ökologische Qualität abzielende Wasserrahmenrichtlinie 2000/60/EG und die Hochwasserrisikomanagementrichtlinie 2007/60/EG sind in engem Zusammenhang zu sehen. Die Erstellung der vorgegebenen Pläne für das Einzugsgebietsmanagement und das Hochwassermanagement sind ergänzend zu betrachten. Sie enthalten umfangreiche technische und nicht-technische Maßnahmenpakete, die das Ergebnis umfassender Abwägungsprozesse sind beziehungsweise sein werden. Der Erfolg der Planumsetzung hängt entscheidend vom Zusammenspiel der technischen und nicht-technischen Maßnahmen ab. Es muss gewährleistet sein, dass in diesem Zusammenhang technische Maßnahmen für die Hochwasserrichtlinie optimal funktionieren.

Seit den siebziger Jahren werden mono-, als auch multikriterielle Verfahren zur Optimierung des Speicherbetriebs und von Hochwasserrückhaltebecken eingesetzt. Zwischenzeitlich haben sich in den Ingenieurwissenschaften neue Optimierungs-

algorithmen etabliert. Eine Klasse dieser relativ neuen Verfahren sind Evolutionäre Algorithmen (EA). Während diese in einigen Bereichen der Wasserwirtschaft bereits Anwendung finden, werden im Bereich der Speicheroptimierung und des Hochwassermanagements meist Methoden der Dynamischen Programmierung (DP) angewendet. Die verschiedenen Klassen von Optimierungsalgorithmen haben ihre jeweiligen Vor- und Nachteile. Daher war es Ziel dieser Arbeit, einen Memetischen Evolutionären Algorithmus (MEA) zu entwickeln und zu erproben, ob dieser zur Optimierung von Speichersystemen zum Zwecke des Hochwasserschutzes geeignet ist. Der Begriff Mem geht auf den griechischen Begriff Μνήμη (Mneme) für „die Erinnerung“ zurück. Der wesentliche Unterschied zu bisherigen Evolutionären Algorithmen ist, dass bei Memetischen Algorithmen zusätzlich die in die biologische Evolution eingebettete kulturelle (memetische) Evolution eingesetzt wird, um die lokale Suche in den globalen Optimierungsprozess zu integrieren.

Es wurde zunächst eine Kombinatorische Evolutionsstrategie (CES) für nominell diskrete Repräsentationen und auf dieser Grundlage ein Memetischer Evolutionärer Algorithmus entwickelt. Der memetische Algorithmus setzt dabei Operatoren für nominell diskrete Parameter als auch Operatoren für reelle Parameter ein und ist für mono- als auch multi-kriterielle Problemstellungen geeignet.

Reellwertige Evolutionsstrategien zur Optimierung von Speichervolumen und Betriebsregeln wurden in vielen Beispielen bereits erfolgreich angewendet. Daher wird in dieser Arbeit zunächst untersucht, ob die rein kombinatorische Optimierung mittels nominell diskreter Parameter zur Optimierung des Speichersystems möglich ist. Dafür wurde eine problemadäquate Repräsentation von Speichersystemen entwickelt, die es ermöglicht Hochwasserspeicher und deren Kombinationen abzubilden und zu optimieren. Auf dieser Grundlage wurden geeignete nominell diskrete Reproduktions-, Mutations- und Selektionsoperatoren erprobt. In einem zweiten Schritt wurden nominell diskrete Operatoren und reellwertige Operatoren zu einem neuen memetischen Algorithmus aufgebaut. Ziel dabei ist es, die Kombination der Speicher als auch der für die Volumina und die Steuerung erforderlichen reellwertigen Parameter zur Dimensionierung in einem Vorgang zu optimieren.

Die Ergebnisse zeigen, dass die nominell diskrete und reelle Optimierung von Speichersystemen sehr gut möglich ist. Mono- und multikriterielle Problemstellungen wurden mit den beiden neuen Algorithmen schnell und zuverlässig optimiert. Das Ergebnis der Optimierung zeigt die vollständige Bandbreite der Handlungsalternativen und ermöglicht a posteriori Entscheidungen. Und unabhängig von der Auswahl eine hohe Sicherheit hat, dass die gewählte Maßnahmenkombination die Hochwasserscheitel effektiv reduziert. Die entwickelte Kombinatorische Evolutionsstrategie als auch der Memetische Algorithmus haben sich als zuverlässiges Werkzeug zur Lösung der hier betrachteten Problemstellungen erwiesen.

Insbesondere die multikriterielle Optimierung zeigt deutlich das Potenzial zur Steigerung der Effizienz von Hochwasserspeichersystemen. Die Optimierungsergebnisse stellen eine Entscheidungshilfe zur Planung von Hochwasserschutzmaßnahmen dar bei welcher der Entscheidungsträger frei aus einer Menge „besten Kompromisse“ wählen kann. Durch dieses Verfahren ist es nicht mehr erforderlich, mehrere reellwertige Optimierungen durchzuführen, da sowohl der nominelle Lösungsraum als auch der reellwertige Lösungsraum zuverlässig untersucht wird.

Schlagerwörter: Hochwasserspeicher, Hochwassermanagement, Hochwasserschutzmaßnahmen, memetische evolutionäre Optimierung, Evolutionsstrategie, genetische Algorithmen, multikriterielle Optimierung.

Abstract

The planning of flood retention reservoirs in regards to volume and control is a complex engineering task, because measures like flood retention basins, dams or renaturation measures effect each other. There are not many degrees of freedom in design and planning within highly populated areas. Therefore, flood-management-projects have to be embedded in an integral planning process. Flood-management concerns not only the technical-economical sectors of society, but also the respective natural and social environment. In addition, climate change could alter the precipitation patterns, which in the future may have crucial influence on flood generation and flood discharge. These kind of external impacts are hard to predict, but within the project planning, which is conducted with an eye on the future, they have to be considered. Altogether, within the flood-management the decision-making processes are getting more complex and more interdisciplinary, not least because of the involvement of the public. It is essential to provide methods and algorithms, which speed up the decision processes and make them transparent.

An important challenge is new directives of the European Community. The European Water Framework Directive 2000/60/EG and the directive 2007/60/EC on the assessment and management of flood risks is to be seen in close conjunction. The development of pre-determined catchment plans and flood management plans have supplementary to be considered. They contain comprehensive technical and non-technical measurement packages, which are or rather will be the result of extensive consideration processes. A successful realization of planning is closely connected to the coherence of technical and non-technical measures. For this reason, it is essential that technical measures for the purpose of the flood-directive perform optimally.

Since the seventies mono- as well as multi-criteria optimization algorithms have been used to optimize the reservoir control of retention basins. Meanwhile new optimization methods are widely established in engineering. One class of these relatively new methods is the Evolutionary Algorithms (EA). Although these algorithms have already been used successfully in some areas of water management, in the field of reservoir and flood management methods based on Dynamic Programming (DP) are used. Each class of algorithms has its pros and cons. Because of that, the purpose of this work has been to develop a Memetic Evolutionary Algorithms (MEA), to test and prove that this kind of algorithm is capable of reser-

voir system optimization for the needs of flood management. The term Mem is based on the Greek term Μνήμη (Mneme) which means “the memory”. The considerable difference from previous Evolutionary Algorithms is that these Memetic Algorithms use the cultural evolution, in addition to genetic evolution, which allows a local search within the global optimization process.

At first Combinatorial Evolution Strategy (CES) for nominal discrete representations has been developed. A Memetic Evolutionary Algorithm has been developed based on the Combinatorial Evolution Strategy. This memetic Algorithm makes use of nominal discrete as well as real operators and can be used to solve mono- as well as multi-criteria problems.

Real value Evolution Strategies for the optimization of retention volume and control rules has been many times used successfully. Because of that, a first investigation has been to analyze whether it is possible to optimize the reservoir system with the pure combinatorial optimization by the use of nominal discrete parameters. Therefore, a representation of the reservoir system has been developed adequate for the problem, in order to represent and optimize reservoirs and their combinations. Based on that the suited nominal discrete reproduction-, mutation-, and selection operators have been evaluated. At the second step, these nominal discrete operators and well-known real value operators have been used to develop a Memetic Algorithm. This allowed optimizing the combination of the reservoirs as well as the real value retention volume and the control parameters within a single optimization process.

The results demonstrate that nominal discrete optimization and at the same time, real value optimization of reservoir systems is highly feasible. Mono- as well as multi-criteria problems can be optimized with the both developed algorithms fast and reliable. The result of the optimization portrays the complete bandwidth of the decision space and moreover allows the so-called a posteriori decision. With this kind of set of solutions, a decision maker can freely choose a solution according to its own preferences, but can always be sure, that the selected combination of measures is able to reduce the flood peak effectively. The developed Combinatorial Evolution Strategy as well as the Memetic Algorithm is reliable tool to solve flood management problems.

Especially the multi criteria optimization demonstrates the significant potential to increase the efficiency of flood reservoir systems. Those optimization results are an essential deci-

sion support for the planning of flood control measures, where the decision maker is able to choose out of an abundance of “best compromises”. With this method, it is not anymore necessary to do several real value optimizations, because the nominal discrete solution space as well as the real value solution space is reliably analyzed.

Keywords: flood retention reservoirs, flood management, flood control measures, memetic evolutionary optimization, evolution strategy, genetic algorithm, multi criteria optimization, mixed integer optimization.

Danksagung

Die vorliegende Arbeit entstand im Wesentlichen während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Fachgebiet für Ingenieurhydrologie und Wasserbewirtschaftung der Technischen Universität Darmstadt.

An dieser Stelle möchte ich mich bei Herrn Prof. Dr.-Ing. Manfred Ostrowski für das mir entgegengebrachte Vertrauen und die Möglichkeit, diese Arbeit anzufertigen zu dürfen, bedanken. Herrn Prof. Dr. Robert Jüpner danke ich für die Übernahme des Korreferates.

Für die zur Verfügung gestellte Datengrundlage aus dem Einzugsgebiet der Erft möchte ich mich beim Erftverband und insbesondere bei Herrn Horst Werner bedanken.

Besonderer Dank gilt auch den Mitarbeitern und Ehemaligen des Fachgebiets, darunter Michael Bach, Dr. Detlev Belke, Jochen Hack, Jan Koch, Dr. Arne Klawitter, Oliver Kraft, Amir Mobasher, Dr. Dirk Muschalla, Frank Reußner, Dr. Kai Schröter und Dr. Axel Winterseid für die konstruktive Zusammenarbeit und fruchtbare Diskussionen und Anregungen.

Judith Feldmann und Harry Koudele möchte ich ganz herzlich für die tatkräftige Unterstützung bei organisatorischen Aufgaben danken.

Bei Isabel Rathgeber bedanke ich mich für die kompetente, zuverlässige und schnelle Hilfe.

Herzlich danken möchte ich meinen Eltern, die durch vielfältige Fürsprache und ihre Unterstützung wesentlich zum Gelingen dieser Arbeit beigetragen haben.

Bei meiner Freundin Dr. Elena Algaer bedanke ich mich ganz herzlich für die entgegengebrachte Rücksichtnahme und Entlastung während des Verfassens dieser Arbeit und für die fruchtbaren Anregungen bei den komplexen mathematischen Problemstellungen.

Darmstadt, im Dezember 2009

Christoph Hübner

Vorwort

Die Planung, der Bau und der Betrieb von Hochwasserschutzanlagen ist ein immerwährendes Problem der Wasserwirtschaft. Über viele Jahrzehnte standen der technische Hochwasserschutz und seine monetäre Bewertung im Vordergrund des Handelns. Im letzten Jahrzehnt ist jedoch eine Hinwendung zu einer integrierten Betrachtung der vielfältigen Nutzungsansprüche an Gewässerkorridore festzustellen. Insbesondere haben sich die Gewässerökologie und die Reduzierung des Schadenspotenzials als weitere Zielsetzungen etabliert. Wesentliche Antriebe zur integrierten Betrachtung gehen von Rahmenrichtlinien der Europäischen Kommission (Hochwasserrichtlinie und Wasserrahmenrichtlinie) aus, die inzwischen in nationales Recht umgesetzt wurden.

Gleichwohl bleibt der technische Hochwasserschutz, insbesondere durch Retentionspeicher in Form von Poldern, Hochwasserrückhaltebecken und Talsperren eine wesentliche Komponente des integrierten Vorgehens. Hochwasserschutzsysteme sind häufig sehr komplexe Systeme, deren Lage und Betrieb möglichst optimal aufeinander abzustimmen sind. Hier setzt die Arbeit von Christoph Hübner an. Der Autor stellt die Hypothese auf, dass mit sog. memetischen evolutionären Optimierungsalgorithmen eine simultane hybride Optimierung sowohl ganzzahliger, d. h. nominell diskreter als auch und kontinuierlicher d. h. reeller Entscheidungsvariablen möglich ist. Hierbei sind unter ganzzahligen Entscheidungsvariablen z. B. die Lage einer Alternative und unter kontinuierlichen Variablen z. B. das Volumen und die Speicherabgaben zu verstehen. Zum Nachweis der Funktionalität des memetischen Algorithmus wird dieser zunächst auf ein bekanntes komplexes Optimierungsproblem und nachfolgend auf eine reale Fallstudie angewendet werden.

Der Autor verfolgt neben der Entwicklung und Test des Algorithmus auch das Ziel, diesen als Softwareinstrument zugänglich zu machen. Dies erfolgt durch Integration des Algorithmus in ein umfassendes wasserwirtschaftliches Planungskonzept, das von mehreren Doktoranden des Fachgebiet IHWB entwickelt wurde (Akronym BlueM).

Simulationsbasierte Suchalgorithmen sind rechenintensiv. Die Anwendung neuartiger Verfahren wie dieses memetischen Algorithmus wären ohne die Fortschritte in der Computertechnologie nicht vorstellbar. Auch diesem Problem geht der Autor nach. Es werden zunächst verschiedene Methoden zur Beschleunigung des Rechenablaufs durch

Parallelisierung diskutiert; die Auswahl der gewählten Multithreading Methode wird begründet und ihre Funktionalität wird nachgewiesen.

Es ist anzumerken, dass in der Praxis eine erhebliche Skepsis gegenüber Optimierungsverfahren festzustellen ist. Woher diese Skepsis rührt, ist nicht eindeutig. Daher möchte ich deutlich darauf hinweisen, dass sich die Entscheidungsträger selbst den Weg zu einer nachvollziehbaren optimalen Entscheidung verbauen. Multikriterielle Verfahren der von Christoph Hübner vorgestellten Art erlauben einen hohen Grad an Transparenz und die nachträgliche Abwägung unter konkurrierenden Zielen. Ich bin in diesem Sinn gespannt darauf, ob diese Verfahren sich in Zukunft auch in der praktischen Anwendung etablieren werden.

Darmstadt, im Juli 2010

Prof. Dr.-Ing. Manfred Ostrowski

Inhaltsverzeichnis

Kurzfassung	I
Abstract	IV
Danksagung	VII
Vorwort	VIII
Inhaltsverzeichnis	XI
Abbildungsverzeichnis	XV
Tabellenverzeichnis	XVIII
Verzeichnis der Algorithmen	XIX
Symbolverzeichnis	XX
1 Einleitung	1
1.1 Veranlassung	2
1.2 Problemstellung	3
1.3 Zielsetzung	4
1.4 Entwicklungsvorhaben	5
1.5 Methodik	6
1.6 Inhalt der Arbeit	7
2 Grundlagen der Optimierung im Hochwasserschutz	11
2.1 Definitionen im Hochwasserschutz	11
2.1.1 Hochwasserschutz und Hochwassermanagement	12
2.2 Optimierung von Speichersystemen – Stand der Wissenschaft	15
2.3 Forschungsbedarf	19
2.4 Grundlagen evolutionärer Algorithmen	20
2.4.1 Klassifizierung der Optimierungsalgorithmen	20
2.4.2 Historische Entwicklung evolutionärer Algorithmen	23
2.4.3 Optimierungsprobleme	26
2.4.4 Mono- und multikriterielle Optimierung	28
2.5 Problemdomäne Hochwasserspeichersystem	31

2.5.1	Zielfunktion	33
2.6	Grundlagen der reellwertigen Evolutionsstrategie	34
2.6.1	Repräsentation	34
2.6.2	Reproduktions- und Mutationsoperatoren	35
2.6.3	Prozedur des reellwertigen Algorithmus	36
2.7	Grundlagen der kombinatorischen Evolutionsstrategie	39
2.7.1	Problemdomäne Handelsreisendenproblem (TSP)	40
2.7.2	Repräsentation und Operatoren	41
2.7.3	Prozedur des Algorithmus	42
2.7.4	Erprobung	44
2.7.5	Diskussion	46
3	Methodik der Hochwasserspeicheroptimierung	49
3.1	Methodik der nominell diskreten Hochwasserspeicheroptimierung	49
3.1.1	Repräsentation der Hochwasserspeicher	49
3.1.2	Reproduktions- und Mutationsoperatoren	53
3.1.3	Prozedur des Algorithmus	54
3.1.4	Erprobung	55
3.1.5	Diskussion	56
3.2	Methodik der Memetischen Hochwasserspeicheroptimierung	58
3.2.1	Voraussetzung zur memetischen Optimierung	60
3.2.2	Reproduktions- und Mutationsoperatoren	61
3.2.3	Selektionsoperator	61
3.2.4	Prozeduren und Repräsentationen der Algorithmen	62
3.2.5	Vergleich und Diskussion	69
3.2.6	Erprobung und Diskussion	72
4	Modell- und Optimierungssystem	73
4.1	Das BlueM Softwarepaket	73
4.2	BlueM.Sim	74
4.3	Das kombinierte Modell und Optimierungssystem	76
4.4	Modellierung des Optimierungsdatensatzes	79

4.5	Performanz des Modell- und Optimierungskonzeptes	80
4.6	Performanz des Modelldatensatzes	81
5	Parallelisierung des Modell- und Optimierungssystems	83
5.1	Methodendiskussion	85
5.2	Grundlagen zur Implementierung	91
5.3	Implementierung der Parallelisierung	92
5.4	Realisierter Geschwindigkeitsgewinn	96
6	Anwendungsbeispiel Erft	99
6.1	Das Einzugsgebiet der Erft	99
6.2	Pegel und Hochwasserereignisse	102
6.3	Bestehende Hochwasserschutzmaßnahmen	104
6.4	Zusätzliche Maßnahmen zur Optimierung	106
6.5	Zielsetzung der Optimierung	111
7	Optimierungsergebnisse	113
7.1	Optimierungsergebnisse des nominell diskreten Algorithmus	113
7.1.1	Monokriterielle nominell diskrete Ergebnisse	113
7.1.2	Multikriterielle nominell diskrete Ergebnisse	120
7.2	Optimierungsergebnisse des memetischen Algorithmus	129
7.2.1	Monokriterielle Ergebnisse des memetischen Algorithmus	129
7.2.2	Multikriterielle Ergebnisse des memetischen Algorithmus	136
7.3	Operatoren und Einstellungen	145
7.3.1	Anpassung der Schrittweite bei der memetischen Optimierung	147
7.3.2	Steuerung der Selektion <i>Emem</i>	148
7.3.3	Lokaler und globaler Lösungsraum	149
8	Fazit und Ausblick	151
	Literaturverzeichnis	157
	Anlagen	167
	Anlage A Repräsentanzen, Operatoren und Verlauf eines Rundreiseproblems	168
	Anlage A - 1 Repräsentanzen des Rundreiseproblems	168
	Anlage A - 2 Reproduktionsoperatoren	170

Anlage A - 3 Mutationsoperatoren	172
Anlage A - 4 Verlauf einer Optimierung des Rundreiseproblems	175
Anlage B Reproduktions- und Mutationsoperatoren zur Lösung des Hochwasserspeicheroptimierungsproblems	176
Anlage B - 1 Reproduktionsoperatoren	176
Anlage B - 2 Mutationsoperatoren	177
Anlage C Klassifizierung des memetischen Algorithmus	179
Anlage D Erftverlegungen	182
Anlage E Hochwasserschutzmaßnahmen	184
Anlage E - 1 Wasserrückhalt und Abflussleistung	184
Anlage E - 2 Hochwasserschutz in der Bauleitplanung	184
Anlage E - 3 Gesetzliche Verankerung des HW Schutz in der Bauleitplanung (Patt und Bechteler, 2001)	185
Anlage F Systemelemente des Erft-Optimierungssystems	186
Anlage G Steuerungsdateien des BlueM.Opt Optimierungswerkzeuges	189
Anlage G - 1 Steuerungsdatei zur kombinatorischen Optimierung	189
Anlage G - 2 Steuerungsdatei mit den Optimierungszielen	190
Anlage G - 3 Steuerungsdatei mit Optimierungsparametern	191
Anlage G - 4 Steuerungsdatei mit Modellparametern	192
Anlage H Einstellungen zur Kontrolle der Algorithmen	193
Anlage H - 1 Steuerparameter des kombinatorischen Algorithmus	194
Anlage H - 2 Steuerparameter des Memetischen Evolutionären Algorithmus	194
Anlage H - 3 Steuerparameter des reellwertigen Algorithmus	194

Abbildungsverzeichnis

Abbildung 2-1: Hochwasserganglinie aus Amt für Wasserwirtschaft (1966)	12
Abbildung 2-2: Klassifizierung der Optimierungsmethoden angelehnt an Syrjakow (2005)	22
Abbildung 2-3: Prinzipieller Ablauf evolutionsstrategischer Algorithmen	25
Abbildung 2-4: Die Qualitäten der Genotypen des Entscheidungsraumes Ω werden durch Simulation ermittelt und im Lösungsraum abgebildet. Dort ist f_1 der Qualitätswert und N die Zahl der Evaluierungen (Simulation und Auswertung der Zielfunktion). In Blau sind die aktuell besten Lösungen dargestellt.	29
Abbildung 2-5: In Blau dargestellt die Lösungen, welche von keiner anderen Lösung dominiert werden.	30
Abbildung 2-6: Bestehende und in dieser Arbeit entwickelte Algorithmen	34
Abbildung 2-7: Beispiel einer multikriteriellen Parameteroptimierung durch die reellwertige Evolutionsstrategie. Die pereto-dominanten Lösungen sind in grün abgebildet.	38
Abbildung 2-8: Zufällig, Gen. 1	44
Abbildung 2-9: Zuf., Gen. 1.000	44
Abbildung 2-10: Zuf., Gen. 19.000	44
Abbildung 2-11: Kreis, Gen. 1	44
Abbildung 2-12: Kreis, Gen. 500	44
Abbildung 2-13: Kreis, Gen. 10.000	44
Abbildung 2-14: Vergleich der erreichten Qualitäten	46
Abbildung 3-1: potenzielle Standorte für Hochwasserspeicher	49
Abbildung 3-2: Logarithmischer Vergleich der Größe der Lösungsräume des Rundreiseproblems (TSP symmetrisch und asymmetrisch) und des hier vorliegenden Hochwasserschutzproblems in Abhängigkeit der Anzahl der Städte bzw. der Standorte	52
Abbildung 3-3: Vergleich der Operatoren für die Problemdomäne Hochwasserschutz	56
Abbildung 3-4: Überlagerung der Lösungsräume	59
Abbildung 4-1: die Teilpakete des BlueM Simulations- und Optimierungssystems (aus Bach u. a. (2009))	74
Abbildung 4-2: Modell und Optimierungssystem (angelehnt an Hättenschwiler und Gachet (2009))	76

Abbildung 4-3: Steuerung des Optimierungsalgorithmus und des Simulationsmodells mithilfe des Controllers	77
Abbildung 4-4: Anbindung der Optimierungsalgorithmen an den Controller. Der memetische Algorithmus setzt Operatoren des kombinatorischen und des reellwertigen Algorithmus ein.	78
Abbildung 4-5: Anbindung der Simulationsmodelle	78
Abbildung 5-1: Geschwindigkeitsgewinn bei der Bearbeitung eines parallelen Problems unter Verwendung von Mehrkernprozessoren bei verschieden großen parallelen Anteilen	87
Abbildung 5-2: Sequenzdiagramm zweier Threads	95
Abbildung 5-3: Anzahl der Simulationen pro Minute durch Parallelisierung der Simulationen bei Verwendung einer 2-Kern-CPU	96
Abbildung 6-1: Einzugsgebiet der Erft (Erftverband, 2008)	100
Abbildung 6-2: Lage der als Eingangsinformation verwendeten Pegel	102
Abbildung 6-3: Abflüsse des Ereignisses Juni 1984 an den 7 Pegeln	103
Abbildung 6-4: Abflüsse des Ereignisses Januar 1995 an den 7 Pegeln	103
Abbildung 6-5: Abflüsse des Ereignisses Oktober 2000 an den 7 Pegeln	104
Abbildung 6-6: Lage der geplanten und in Betrieb genommenen Rückhalteräume	105
Abbildung 6-7: In der Optimierung zusätzlich betrachtete Standorte für Hochwasserspeicher	107
Abbildung 6-8: Optimierungssystemplan der Erft mit 9 Standorten (S1 bis S9)	109
Abbildung 7-1: Monokriterielle nominell diskrete Optimierung mit dem Ziel der Reduzierung des maximalen Abflusses am Zielpiegel	115
Abbildung 7-2: Optimierungsverlauf zur Reduzierung des Abflussscheitels, Anzahl der Simulationen 200, absolutes Optimum nach 132 Simulationen erreicht.	117
Abbildung 7-3: Beste Maßnahmenkombination bei nominell diskreter Optimierung	119
Abbildung 7-4: Multikriterielles nominell diskretes Ergebnis zur Reduzierung des Hochwasserscheitels und des Retentionsvolumens	121
Abbildung 7-5: Datenpunkte und Trendlinie der 41 Hochwasserrückhaltebecken	124
Abbildung 7-6: Optimierung zweier nicht konkurrierender Ziele	127
Abbildung 7-7: Beispiel eines eindeutig bereichsweise funktionalen Zusammenhangs zwischen den Zielfunktionen	128
Abbildung 7-8: MEA Optimierungsverlauf mit einer Zielsetzung	131
Abbildung 7-9: Vergleich der Gradientenwege bei monokriteriellen MEA Optimierungsverläufen.	133

Abbildung 7-10: Memetisches Ergebnis bei der Optimierung Lage, der Volumina und der Abgaben	134
Abbildung 7-11: Abflüsse ohne Maßnahmen	135
Abbildung 7-12: Abflüsse in Folge der besten Lösung	135
Abbildung 7-13: Abflüsse in Folge der Referenzkombination	135
Abbildung 7-14: MEA Optimierung des HW-Scheitels und des Retentionsvolumens	138
Abbildung 7-15: MEA Optimierung des HW-Scheitels und der Investitionskosten	138
Abbildung 7-16: Ausschnitt aus der Memetisch Evolutionären Optimierung	139
Abbildung 7-17: Zuflüsse der Seriensimulation mit 3 Ereignissen	140
Abbildung 7-18: Darstellung des Optimierungsergebnisses mit drei von vier Zielfunktionen	141
Abbildung 7-19: Scatter-Plot-Matrix des Optimierungsergebnisses	142
Abbildung 7-20: Vergrößerung des optimalen Bereiches	143
Abbildung 7-21: Abflussganglinien am Zielpegel der vier selektierten Lösungen	144
Abbildung 7-22: Abflussganglinien des "besten" Ereignisses an verschiedenen Beobachtungspunkten	144
Abbildung 7-23: Optimierungsverlauf bei starker Reduzierung der reellen Schrittweite	147
Abbildung 7-24: Kontinuierlicher Dn Verlauf	148
Abbildung 7-25: Dn Verlauf mit abrupter Änderung	148
Abbildung 7-26: Vergleich der Settings für den $Emem$ Operator	149
Abbildung 7-27: Lokale reelle Lösungsräume	150
Abbildung 8-1: Erftverlegungen (Pflug und Schneider, 1998)	183

Tabellenverzeichnis

Tabelle 2-1: Beispiele für differenzierte Maßnahmen zur Hochwasservorsorge bei unterschiedlichen	14
Tabelle 2-2: Erste Arbeiten zur Optimierung von Speichersystemen	15
Tabelle 2-3: Speicheroptimierung mittels Dynamischer Programmierung	16
Tabelle 2-4: Speicheroptimierung mittels Evolutionärer Algorithmen	18
Tabelle 2-5: Operatoren des reellwertigen, evolutionsstrategischen Algorithmus	35
Tabelle 4-1: Systemelemente des BlueM.Sim Simulationsmodells (Ostrowski et al., 2002; 2009)	75
Tabelle 4-2: Schematischer Schaltplan für die kombinatorische Steuerung des Modelldatensatzes	79
Tabelle 6-1: Gewässersteckbrief (Ministerium fuer Umwelt und Naturschutz, Landwirtschaft und Verbraucherschutz NRW, 2009)	101
Tabelle 6-2: Betrachtete Hochwasserwellen	104
Tabelle 6-3: Bestehende und geplante Hochwasserrückhaltebecken (Erftverband, 2006; 2007; 2008; 2009)	106
Tabelle 6-4: Auszug aus der Steuerungsdatei zur Kontrolle der nominelle diskreten Anteile	110
Tabelle 6-5: Übersicht über die Speicher und deren Abgaben (Min,Max) und Volumina (Min,Max)	111
Tabelle 7-1: Beste Maßnahmenkombination des monokriteriellen nominell diskreten Optimierungslaufes mit dem Ziel den Scheitelabfluss am Zielpegel zu reduzieren	116
Tabelle 7-2: Beste Maßnahmenkombination des monokriteriellen nominell diskreten Optimierungslaufes mit dem Ziel den Scheitelabfluss an mehreren Positionen zu reduzieren	118
Tabelle 7-3: Auswahl aus den paretooptimalen Lösungen	122
Tabelle 7-4: Auswahl aus den paretooptimalen Lösungen	126

Verzeichnis der Algorithmen

Algorithmus 2-1: Prozedur der parametrischen Evolutionsstrategie	37
Algorithmus 2-2: Entwickelter Algorithmus zur Rundreiseoptimierung	43
Algorithmus 3-1: Entwickelter evolutionsstrategischer kombinatorischer Algorithmus für nominell diskrete Zahlen	55
Algorithmus 3-2: Sequenzielle Prozedur des MEA	64
Algorithmus 3-3: Parallele Prozedur des MEA mit allen reellen Parametern	65
Algorithmus 3-4: Parallele Prozedur mit selektierten reellen Parametern	67
Algorithmus 5-1: Prozedur der Parallelisierung der Simulationen innerhalb jeder Generation	93

Symbolverzeichnis

Symbol / Akronym	Dimension / Menge	Bezeichnung
C		Investitionskosten
D	$D_i = \{d_{i,1}, \dots, d_{i, D_i }\}$	Nomineller Entscheidungsraum
D_i		Menge der diskreten Zahlen
D_n		Schrittweite
d		Nominal diskrete Variablen
\vec{d}	$\in D_i$	Nominal diskreter Entscheidungsvektor
$\vec{d}(k)$		Vektor der nominalen Entscheidungsvariablen eines Elter in Abhängigkeit der reellen Variablen
\vec{d}^*		Reproduzierter nomineller Entscheidungsvektor
\vec{d}^{**}		Mutierter nomineller Entscheidungsvektor
f		Sequenzieller Anteil eines Algorithmus
$f(x)$		Evaluierungsfunktion
$g(x)$		Randbedingungen
k		Anzahl der Ziele
l		Länge einer Tour
m		Anzahl der Randbedingungen
\min		Minimierungsziel
n		Anzahl der Städte
P		Population
P_0		Startpopulation
P_{Mem}		Memory Population
P_N		Nominaler Teil der Population
P_R		Reeller Teil der Population
P_i		Population der i -ten Generation
P_i^*		Reproduzierte Population der i -ten Generation
P_i^{**}		Mutierte Population der i -ten Generation
P_i^{***}		Evaluierte Population der i -ten Generation
p		Anzahl der Prozessoren
Q	$[m^3/s]$	Abfluss

Q_{max}	$[m^3/s]$	Abflussscheitel
\mathbb{R}		Menge der reellen Zahlen
r		Kontinuierliche Variablen
\vec{r}	$\in [r_i^{min}, r_i^{max}] \subset \mathbb{R}$	Reeller Entscheidungsvektor
$\vec{r}(m)$		Vektor der reellen Entscheidungsvariablen in Abhängigkeit der nominalen Variablen
\vec{r}^*		Reproduzierter reeller Entscheidungsvektor
\vec{r}^{**}		Mutierter reeller Entscheidungsvektor
$S_p(n)$		Potenzieller Geschwindigkeitsgewinn
T		Anzahl der möglichen Touren
$T'(n)$		Zeit zur sequenziellen Ausführung eines Programmteils
$T_p(n)$		Zeit zur parallelen Ausführung eines Programmteils
V_R		Retentionsvolumen
w		Wichtungsfaktor
X		Reproduktionsoperator
$X(\mu)$		Reproduktionsoperator
X_N		Natürlicher Reproduktionsoperator
X_R		Reeller Reproduktionsoperator
\vec{x}	$\in \Omega$	Entscheidungsvektor
\vec{y}		Ergebnisvektor
\mathbb{Z}		Menge der ganzen Zahlen
Z		Optimierungsziel
z		Ordinal diskrete Variablen
\vec{z}	$\in [z_i^{min}, z_i^{max}] \subset \mathbb{Z}$	Ordinal diskreter Entscheidungsvektor
γ		Anzahl der Generation
θ		Mutationsoperator
θ_N		Natürlicher Mutationsoperator
θ_R		Reeller Mutationsoperator
Λ		Lösungsraum
λ		Anzahl der Nachfahren

μ		<i>Anzahl der Eltern</i>
\mathcal{E}		<i>Selektionsoperator</i>
\mathcal{E}_M		<i>Multikriterieller Selektionsoperator</i>
\mathcal{E}_{Mem}		<i>Memory Selektionsoperator</i>
ϕ	$\Omega \rightarrow \Lambda$	<i>Evaluierungsfunktion</i>
\aleph		<i>Mächtigkeit von \mathbb{R}</i>
Ω		<i>Entscheidungsraum</i>

1 Einleitung

Hochwasser sind Naturereignisse, denen der Mensch immer ausgesetzt war und auch künftig immer ausgesetzt sein wird. Ein Hochwasser ist zunächst noch keine Katastrophe. Es wird erst dann zur Katastrophe, wenn es sich zu „einem Ereignis im Raum konzentriert, bei dem eine Gesellschaft einer schweren Gefährdung unterzogen wird“ (Plate und Merz, 2001). Schäden und Gefahren für den Menschen entstehen erst dann, wenn Hochwasser anthropogen genutzte Flächen überfluten, wie zum Beispiel besiedelte oder landwirtschaftlich genutzte Gebiete. In diesen Gebieten können im Falle eines Hochwassers sowohl der Verlust von Menschenleben oder materielle Schäden eintreten, sodass "die lokale gesellschaftliche Struktur versagt und alle oder einige wesentliche Funktionen der Gesellschaft nicht mehr erfüllt werden können" (ebd.).

Laut International Emergency Disasters Database der WHO (EM-DAT, 2010) sind 10 % aller durch Naturkatastrophen bedingten Todesfälle innerhalb der letzten 30 Jahre auf Überschwemmungen zurückzuführen. Und weltweit steigt die Zahl der Überschwemmungen. Die EM-DAT verzeichnet weltweit (Stand: 08.01.2010) zwischen 2000 und 2009 1654 Überschwemmungen, 886 Mio. Betroffene, 159 Mrd. US\$ Schäden und 52.457 Verluste an Menschenleben. Diese machen immense Investitionen in Schutzvorkehrungen notwendig. In einer Auswertung des mittleren Schadensrisikos aus Naturgefahren in der Schweiz wurde ermittelt, dass das Risiko ca. 15 Mrd. Franken pro Jahr beträgt. Etwa 15 % dieses Risikos ist Hochwasser zuzuschreiben. Der übrige Anteil verteilt sich auf Erdbeben, Sturm, Lawinen und andere Gefahren. Allerdings werden zur Hochwasservorsorge in der Schweiz 56 % der gesamten Ausgaben der öffentlichen Hand für die Vorsorge gegen Naturereignisse eingesetzt. Natürlich muss berücksichtigt werden, dass beispielsweise Maßnahmen zur Vorbeugung gegen Erdbeben in erster Linie durch die Betroffenen in Form von Bauvorsorge finanziert werden müssen und nicht durch den Staat. Die Tendenz der Verschiebung der Mittelaufwendung für den Hochwasserschutz dürfte aber bleiben (Bundesamt für Zivilschutz, Bern, 1995 aus Patt und Bechteler, 2001).

Die hohe ästhetische und wirtschaftliche Attraktivität von Flächen in der Nähe von Gewässern und die dadurch bedingte Bautätigkeit in Nahbereichen von Überschwemmungsgebieten erfordern ein Hochwassermanagement, das die Balance zwischen Hochwasserschutz, Umweltschutz, Ökonomie und sozialen Belangen erhält oder herstellt.

Diese divergierenden Ziele steigern die Komplexität der Planungs- und Entscheidungsprozesse bei der Entwicklung von Hochwasserschutzkonzepten, sind aber zentraler Ansatz der nachhaltigen Nutzung natürlicher Ressourcen (Dietrich, 2006). Young (1992) bezeichnet die drei übergeordneten Ziele mit:

- Environmental integrity
- Economic efficiency
- Equity

Dietrich (2006) übersetzt dies mit *Unversehrtheit der Umwelt, ökonomische Effizienz* und *soziale Gerechtigkeit*, unter Berücksichtigung ökonomischer, sozialer und kultureller Aspekte. Diese integrale Betrachtung und das in der Gesellschaft gestiegene Umweltbewusstsein beeinflusst auch die Sicht auf den Ausbau sowie die Unterhaltung von Fließgewässern und stellt neue und veränderte Anforderungen an die Wasserwirtschaft. Nicht mehr nur funktionierende technische Lösungen für Nutzungszwecke werden erwartet, sondern auch die Abschätzung und Bewertung der ökologischen und wirtschaftlichen Folgen (Lüderitz und Jüpner, 2009).

Nachhaltiges Handeln muss künftig auch den Klimawandel im Hochwassermanagement berücksichtigen. Damit verbunden ist mit einer Veränderung des Niederschlagsregimes hin zu Extremen zu rechnen, was einen entscheidenden Einfluss auf die Hochwasserentstehung haben kann (Bultot, 1988; Bultot et al., 1992; Arnell und Reynard, 1996; Gellens und Roulin, 1998 zitiert nach: Bronstert et al., 2001).

1.1 Veranlassung

Erwartete Klimaveränderungen und deren Folgen fordern einen vorausschauenden Umgang mit unseren Gewässerkorridoren. Es ist mit höheren Bemessungsabflüssen im Hochwasserfall und deren häufigerem Auftreten zu rechnen. Der Umgang mit diesen neuen Anforderungen verlangt komplexe Analysen und Synthesen der Handlungsalternativen. So wurde zur Kompensation eines mit dem Klimawandel verbundenen Wasserspiegelanstieges der Maas in den Niederlanden ein „Baukasten Maas“ entwickelt, der Maßnahmen nach dem Prinzip „Raum für den Fluss“ unter hydraulischen, raumordnerischen und zukunftsorientierten Aspekten ganzheitlich zu Maßnahmenpaketen zusammenstellt (Schielen und Barneveld, 2006). Dieser sollte die Frage beantworten, inwieweit die technisch möglichen

Hochwasserschutzmaßnahmen in den Niederlanden und bei den Oberliegern im benachbarten Ausland die zunehmenden Wasserstände aufgrund der Klimaveränderung kompensieren können (ebd.). Dazu wurde mit dem hydraulischen Modell Sobek (Deltares, 2009) der Effekt aller im Baukasten enthaltenen Maßnahmen auf die Wasserstände der Maas berechnet. Neben hydraulischen Effekten wurden für alle Maßnahmen auch die Konsequenzen für andere Funktionen bestimmt, wie beispielsweise für Natur oder Erholung (ebd.). Darüber hinaus wurden die Kosten aller Maßnahmen prognostiziert. Die hydraulischen Effekte werden jedoch im Vorfeld durch eine Simulation für jede Maßnahme individuell ermittelt. Die Ergebnisse dieser Simulationen für über einhundert Maßnahmen wurden in einer Datenbank gespeichert. Diese Datenbank bildete den Kern des Analyse-systems, mit dem Pakete aus einzelnen Maßnahmen zusammengestellt werden können (Vriend und Dijkman, 2003). Daher benötigt der erstellte Baukasten keine weiteren hydraulischen Berechnungen, sondern enthält lediglich ein Additionsverfahren, durch das die kombinierte Wirkung der Maßnahmen überschlägig ermittelt wurde. Da dieses Additionsverfahren kaum Rechenzeit benötigt, konnte der Baukasten in Diskussionen mit Anliegern eingesetzt werden. Der Preis ist jedoch die Ungenauigkeit der Wasserspiegellage. Jeder der Teilnehmer hatte die Möglichkeit mit dem Baukasten zu versuchen, die zunehmenden Wasserstände zu kompensieren, wobei er die Maßnahmen entsprechend seiner eigenen Präferenzen auswählen konnte.

1.2 Problemstellung

Obwohl bei einem großen Gewässer wie der Maas, mit einem mittleren Abfluss (MQ) von $400 \text{ m}^3/\text{s}$, ein Additionsverfahren im Mittel sicher eine erste Näherung liefert, kann unter Umständen der tatsächliche Wasserstand einer gewählten Maßnahmenkombination erheblich abweichen. Jede gewählte Kombination muss im Nachgang hydraulisch berechnet werden. Außerdem ist ein solches Verfahren an kleineren Flüssen kaum möglich, da dort die Wechselwirkungen der Maßnahmen nicht berücksichtigt werden können.

Unter Umständen können Maßnahmen zum Schutz vor Hochwasser negative hydraulische Effekte haben, beispielsweise wenn ein Hochwasserrückhaltebecken die Hochwasserwelle verzögert und sich die Abflüsse überlagern. Außerdem stellt sich die Frage, welche Kombination unter den gegebenen Randbedingungen der Anlieger die Beste ist. Bei über 100 möglichen Maßnahmen wird zwar die händisch ermittelte Kombination das gesetzte

Ziel grundsätzlich erfüllen, aber es existieren mit Sicherheit Maßnahmenkombinationen, welche die gesetzten Ziel besser erreichen und auch bei verschiedenen Szenarien eine robuste Lösung darstellen.

1.3 Zielsetzung

Moderne Simulationsmodelle und Optimierungsverfahren erlauben es, bestehende und künftige Hochwasserspeichersysteme zu optimieren, ohne die Wirkungszusammenhänge und nicht-technische Aspekte zu vernachlässigen. Durch die Berücksichtigung mehrerer Zielfunktionen im Optimierungsprozess muss der Entscheider die Präferenzen seiner Ziele nicht mehr vor der Analyse der Planungsvarianten angeben. Vielmehr wird es ermöglicht, aus einem Pool von sogenannten paretooptimalen Lösungen a posteriori frei nach Präferenz (Gewichtung der Kriterien) auszuwählen. Die Entscheidung ist nicht mehr limitiert durch eine Vorabauswahl von wenigen (mit dem Modell) untersuchten Planungsvarianten.

Es ist gängige Praxis zur Lösung einer ingenieurtechnischen Problemstellung, Varianten zu entwickeln. Wurde in der Vergangenheit das Ziel des Hochwasserschutzes in Form technischer Hochwasserschutzmaßnahmen und deren Steuerung eindeutig präferiert, so ist es heute erforderlich, Mehrfachzielsetzungen (ökologische, ökonomische und raumplanerische Aspekte) in den Optimierungsprozessen zu berücksichtigen. Selbst bei einer multikriteriellen Bewertung von Varianten, beispielsweise mittels Nutzwertanalyse, Analytischer Hierarchischer Prozesse (AHP) oder dem Promethee-Verfahren erfolgen Entscheidungen nur unter beschränkter Rationalität, denn der Raum möglicher Lösungen wird hier nur punktuell betrachtet.

In der Praxis ist derzeit noch eine gewohnheitsbedingte Ablehnung gegenüber Optimierungsalgorithmen zu beobachten. Bei langjähriger Ingenieurserfahrung sind händisch ermittelte Lösungen in vielen Fällen auch tatsächlich sehr gute Lösungen. Allerdings ist es nicht möglich, händisch entworfene Varianten abschließend zu beurteilen und im multikriteriellen Zielraum einen fairen Vergleich der Varianten zu erreichen. Sollen beispielsweise ein oder mehrere Hochwasserspeicher einem bestehenden Verbund aus Hochwasserrückhaltebecken hinzugefügt werden, stellt sich die Frage nach der Lage, der

Größe und der Steuerung des neuen Speichers. Diese Fragen sind mit wenigen Modellrechnungen und Alternativenuntersuchungen nicht mehr zu beantworten.

Die in dieser Arbeit entwickelte Methode erlaubt es dagegen, beispielsweise für ein neues Hochwasserrückhaltebecken in einem existierenden Verbundsystem, die Lage, Größe und die Steuerung zu bestimmen. Dabei wird die Interaktion mit den bestehenden Hochwasserrückhaltebecken vollständig berücksichtigt. Mit dieser Methode ist es möglich, technische Maßnahmen (Rückhaltebecken, -flächen, Deiche, Polder, Deichrückverlegung, Querschnittsaufweitung) und strukturelle Maßnahmen (Renaturierung, verbesserte Infiltration, Entsiegelung, dezentraler Rückhalt, Deichrückverlegung, Querschnittsaufweitung, Deiche) im Verbund zu betrachten und deren Wirkungszusammenhänge zu analysieren. Die Planung muss nicht mehr auf den Vergleich einzelner Varianten beschränkt sein. Mit dieser Methode wird stets die volle Bandbreite von Handlungsalternativen berücksichtigt und Planer werden durch detaillierte Informationen in Form von Optimierungsergebnissen darin unterstützt, den Entscheidungsträgern und der beteiligten Öffentlichkeit optimale Lösungen anbieten zu können. Stellt sich heraus, dass einzelne Maßnahmen seitens der Anlieger nicht gewünscht sind, können diese von der Betrachtung ausgeschlossen werden. Die Diskussion mit Beteiligten beschränkt sich nicht nur auf das Pro oder Kontra einzelner Varianten. Die Methode erlaubt es auf die Belange der Beteiligten während der Planung flexibel zu reagieren.

1.4 Entwicklungsvorhaben

In dieser Arbeit werden Hochwasserspeicher hinsichtlich ihrer Lage, Größe und ihres Betriebs optimiert. Zu diesem Zweck wird ein sogenannter memetischer Algorithmus sowohl zur mono- als auch multikriteriellen, als auch zur nominell diskreten und reellen Optimierung von Hochwasserspeichersystemen entwickelt. Das Ergebnis der Optimierung muss die vollständige Bandbreite der Handlungsalternativen aufzeigen. Diese Methode wird es ermöglichen, eine große Zahl von Maßnahmen hinsichtlich ihres Beitrags zum Hochwasserschutz zu untersuchen, ohne die Wechselwirkung der Maßnahmen während des Hochwassergeschehens zu vernachlässigen. Im Gegensatz zu einem Vergleich einzelner Alternativen können hiermit optimierte Lösungen innerhalb des gesamten Entscheidungsraums ermittelt werden.

Die Notwendigkeit zur Optimierung ist offensichtlich, jedoch werden die Methoden nur sehr sporadisch angewendet. Ein entscheidender Vorteil ist dann vorhanden, wenn bestehende Modelle zum Zwecke der Optimierung eingesetzt werden können. Derartige Modelle liegen vielfach bei Betreibern von Hochwasserschutzsystemen vor. Die hier entwickelten Algorithmen erlauben die Anbindung nahezu beliebiger Simulationsmodelle und bieten damit größtmögliche Flexibilität, aber auch Effizienz.

Die Innovation dieser Arbeit liegt vor allem in der Entwicklung und Anwendung des Memetischen Evolutionären Algorithmus (MEA). Die Bezeichnung „memetisch“ geht auf den Begriff Mem (griechisch Μνήμη „die Erinnerung“) zurück. Der wesentlicher Unterschied zu bisherigen evolutionären Algorithmen ist, dass bei memetischen Algorithmen zusätzlich die in die biologische Evolution eingebettete kulturelle (memetische) Evolution eingesetzt wird. Diese memetische Evolution erlaubt einem Individuum sich Wissen, Erfahrungen und Erinnerungen seines Kulturkreises anzueignen und das Wissen zu erweitern. Der evolutionäre Prozess beinhaltet als zwei Stufen, bestehend aus der biologischen Evolution bei der Geburt und der memetischen Evolution nach der Geburt. Dieser zweite Evolutionsschritt wird in dieser Arbeit für die lokale Suche im globalen Optimierungsprozess eingesetzt.

Mithilfe dieses Algorithmus wird es möglich sein, die vorliegende Problemstellung in einer vertretbaren Rechenzeit zu ermitteln. Durch eine Parallelisierung der Simulation wird die Rechenzeit zusätzlich reduziert. Dabei werden, alle durch das Simulationsmodell abbildbaren Maßnahmen und deren Parameter berücksichtigt, um ein Höchstmaß an Flexibilität für die Praxis zu erreichen.

1.5 Methodik

Um obiges Ziel zu erreichen, wird zunächst eine nominell diskrete Evolutionsstrategie für ganzzahlige Repräsentanzen entwickelt und erprobt. Zu diesem Zweck werden Ganzzahloperatoren und ein entsprechender Algorithmus implementiert. Danach wird der memetische Algorithmus entwickelt, der die Operatoren der nominell diskreten Evolutionsstrategie und einer vorhandenen reellwertigen Evolutionsstrategie verwendet.

Um die aktuelle Entwicklung der Prozessoren in Richtung Mehrkernprozessor nutzen zu können und die erforderliche Rechenzeit zur Optimierung in einem praktikablen Rahmen

zu halten, wird die Evaluierung parallelisiert. Zur Erprobung der Algorithmen und der Parallelisierung wurde ein für die reelle als auch nominell diskrete Optimierung geeigneter Modelldatensatz zur Simulation des Hochwasserwellenablaufs der Erft erstellt. Mit diesem wird die Funktionalität der entwickelten Algorithmen nachgewiesen, das bestehende Optimierungspotenzial im Einzugsgebiet der Erft aufgezeigt und die Charakteristika memetischer Optimierungen diskutiert.

1.6 Inhalt der Arbeit

Kapitel 1 erläutert im Wesentlichen die Problemstellung, die Zielsetzung und die Methodik der Arbeit.

In **Kapitel 2** werden die Grundlagen, auf denen die Entwicklung der Algorithmen aufbaut, erläutert. Nach der Definition wichtiger Begriffe des Hochwasserschutzes werden bisherige Arbeiten zur Optimierung im Hochwasserschutz vorgestellt. Ausgehend von dem dort beschriebenen Kenntnisstand wird der Forschungsbedarf ermittelt. Für das Verständnis der Algorithmen wird zunächst eine Klassifizierung vorgenommen und die grundsätzliche Funktionsweise evolutionärer Algorithmen erläutert. Für die Entwicklung eines Optimierungsalgorithmus muss zunächst die vorliegende Problemdomäne bestimmt werden. Daher wurde ein Fragenkatalog erstellt, der die erforderlichen Punkte enthält, um ein Problem mittels evolutionärer Algorithmen zu lösen. Diese Punkte werden in der Arbeit mehrfach thematisiert. Begonnen wird dabei bereits am Ende des Kapitels 2, um die Grundlagen einer reellwertigen Evolutionsstrategie und einer kombinatorischen Evolutionsstrategie zu erläutern. Zur Demonstration wird ein rein mathematisches Problem verwendet, um erstens für die Hochwasserspeicheroptimierung anwendbare Operatoren und Prozeduren zu untersuchen und zweitens den softwareseitigen Rahmen, an den später das Simulationsmodell angebunden wird, zu entwickeln. Die mathematische Problemstellung ist das Problem des Handelsreisenden. Es wurde bewusst gewählt, da sie eine Stufe komplexer ist als die hier vorliegende Problematik des Hochwasserschutzes. So konnte sicher gestellt werden, dass der neu entwickelte nominell diskrete Algorithmus den Anforderungen hinsichtlich der Zuverlässigkeit und Geschwindigkeit genügt.

In **Kapitel 3** wird zunächst der nominell diskrete (kombinatorische) Algorithmus zur Hochwasserspeicheroptimierung beschrieben, um zunächst Lage und Typ der Maßnahme

optimieren zu können. Im Anschluss wird der entwickelte memetische Algorithmus zur Optimierung der Lage und des Typs der Speicher als auch des erforderlichen Retentionsvolumens klassifiziert und beschrieben. Nach der Klassifizierung des Algorithmus werden drei Prozeduren vorgestellt, um die sogenannte gemischt-ganzzahlige Optimierung zu ermöglichen. Dabei werden keine neuen Reproduktions- und Mutationsoperatoren eingeführt. Der memetische Algorithmus setzt die bereits vorhandenen Reproduktions- und Mutationsoperatoren ein. Nach der Diskussion von 3 geeigneten memetischen Prozeduren wird ein eigens entwickelter Selektionsoperator vorgestellt, welcher eine der wesentlichen Eigenschaften des memetischen Algorithmus darstellt. Für den nominell diskreten und den memetischen Algorithmus werden die Punkte des Fragenkatalogs je nach Relevanz bearbeitet. Die Funktionsfähigkeit der Prozedur und der Operatoren des nominell diskreten Algorithmus werden in 3.1 erprobt und die Ergebnisse diskutiert. Die Erprobung und die Demonstration der Funktionsfähigkeit des memetischen Algorithmus erfolgt abschließend am Anwendungsbeispiel Erft in Kapitel 7.

Kapitel 4 stellt das verwendete Modell- und Optimierungssystem vor. Zur Simulation des Hochwassergeschehens wird das Softwarepaket BlueM verwendet. Die Evaluierung durch das Simulationsmodell benötigt mit Abstand am meisten Rechenzeit im gesamten Optimierungsprozess. Das Kapitel endet mit der Performanz des Modell- und Optimierungssystems und des Modelldatensatzes.

Kapitel 5 beschreibt die Parallelisierung des Optimierungsprozesses um die Rechenzeit zu verkürzen. Dabei werden die verschiedenen Methoden zur Parallelisierung detailliert vorgestellt und im Anschluss diskutiert. Nach der Beschreibung der Implementierung der Parallelisierung werden die realisierten Geschwindigkeitsgewinne vorgestellt.

Im **Kapitel 6** wird das Anwendungsbeispiel Erft vorgestellt. Nach der Beschreibung des Gewässers werden die bestehenden Hochwasserschutzmaßnahmen beschrieben. Im Anschluss werden die zusätzlichen Maßnahmen und der Systemplan vorgestellt.

Kapitel 7 demonstriert die Funktionsfähigkeit und Leistungsfähigkeit der Algorithmen bei der Optimierung der Maßnahmen im Einzugsgebiet der Erft. Im ersten Teil des Kapitels werden mono- und multikriterielle Ergebnisse des nominell diskreten Algorithmus vorgestellt und im zweiten Teil mono- und multikriterielle Ergebnisse des memetischen

Algorithmus. Im dritten Teil werden Steuerparameter und die Thematik „Optimierung der Optimierung“ diskutiert.

Die Arbeit schließt mit dem **Kapitel 8** „Fazit und Ausblick“.

Die **Anlagen** enthalten detaillierte Beschreibungen der Repräsentanzen und Operatoren (Anlage A und Anlage B), die Klassifizierung des memetischen Algorithmus (Anlage C), eine Übersicht über Hochwasserschutzmaßnahmen im Allgemeinen (Anlage E) und Informationen Steuerungsdateien und Steuerungsparametern des Algorithmus.

2 Grundlagen der Optimierung im Hochwasserschutz

Das Kapitel gibt zunächst einen Überblick über vergangene und aktuelle Entwicklungen im Bereich des Hochwasserschutzes. Zugleich werden die Grundlagen gelegt, die zur Lösung und zum Verständnis der vorliegenden Problemstellung erforderlich sind. Dabei wird eine Klassifizierung von Optimierungsverfahren vorgenommen, und die Prinzipien evolutionärer Algorithmen werden kurz erläutert.

2.1 Definitionen im Hochwasserschutz

Hochwasser ist das zeitlich begrenzte Anschwellen des Durchflusses über den Basisdurchfluss, das einen für den jeweiligen Durchflussquerschnitt bestimmten Grenzwert, beispielsweise den Ausuferungsdurchfluss, überschreitet (Amt für Wasserwirtschaft (1966), Dyck (1980a) sowie Dyck und Peschke (1995)). Hochwasser können mit völlig unterschiedlichen Charakteristika auftreten. Die Extreme wären hier „Sturzfluten“, die vermehrt in Gebirgen auftreten und innerhalb weniger Minuten alles mit sich reißen, bis hin zu den über Wochen anschwellenden Hochwasserfluten großer Ströme, die vor allem durch die lang andauernde Durchfeuchtung des Untergrunds und der Hochwasserdeiche gefährden (Patt und Bechteler, 2001).

Die maßgeblichen Kenngrößen eines Hochwassers sind **Scheitel** (Durchflussmaximum), **Fülle** (auch Volumen, Durchflusssumme oder Abflusssumme), **Dauer** und **Form** (Dyck und Peschke, 1995; Bronstert et al., 2001; Klein et al., 2008). Das Hochwassergeschehen kann für einen Gerinnequerschnitt mit Hochwasserganglinien beschrieben werden. Wie in Abbildung 2-1 ersichtlich, sind die wesentlichen Elemente der Ganglinie der **Hochwasseranstieg**, der **Hochwasserscheitel** und der **Hochwasserabstieg**.

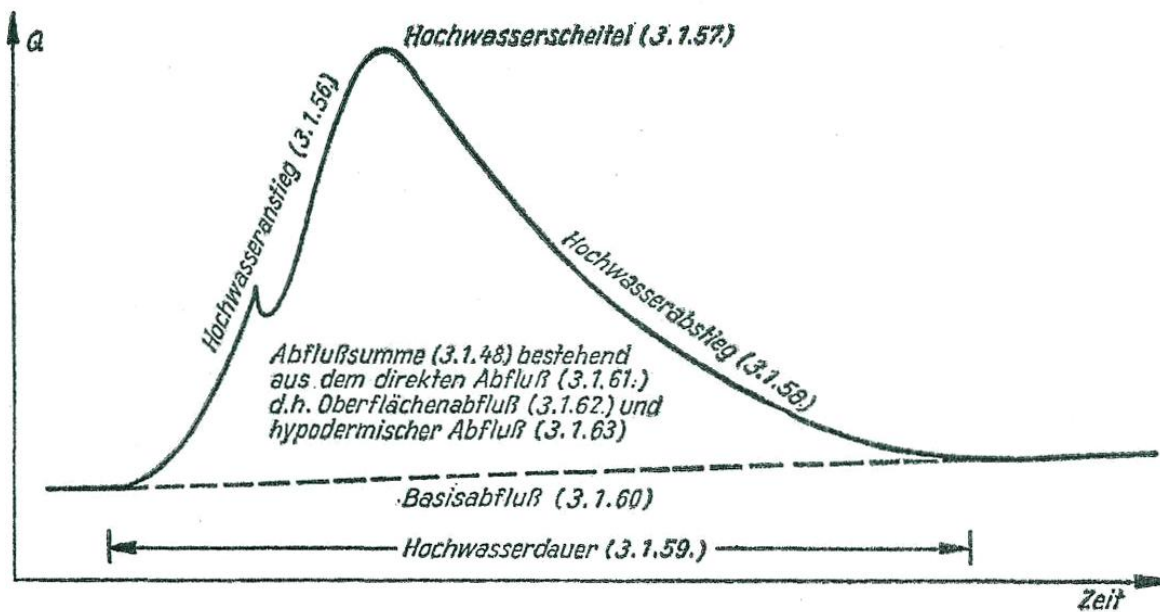


Abbildung 2-1: Hochwasserganglinie aus Amt für Wasserwirtschaft (1966)

Aufgrund der Retention im Flussbett und in der Aue wird die Hochwasserwelle gedämpft und verändert daher während ihres Ablaufs ihre Form. Die Dauer, die ein Wellenscheitel braucht, um von A nach B zu gelangen (ihre Laufzeit) ermöglicht eine Vorhersage, zu welcher Zeit Hochwasserwellen an bestimmten Orten eintreffen und wie hoch die Wasserstände werden.

2.1.1 Hochwasserschutz und Hochwassermanagement

In Publikationen zum Thema Hochwasserschutz fällt auf, dass sich im Laufe der Jahre neben einem inhaltlichen Wandel auch ein begrifflicher Wandel vollzogen hat. Während früher Begriffe wie „Hochwasserschutz“ sowie „Hochwasserschutzpläne“ Gebrauch fanden, werden heute vorzugsweise die Begriffe „Hochwasserrisikomanagement“ sowie „Hochwasserrisikomanagementpläne“ verwendet. Auch in internationalen Veröffentlichungen wird vermehrt von „Flood Risk Management“ statt „Flood Protection“ oder „Flood Control“ gesprochen. Ein Beleg für diesen begrifflichen Wandel ist in der letzten Abschlusserklärung des renommierten „International Symposium on Flood Defence (ISDF)“ zu finden. Dort wurde erklärt, dass der Name der Konferenz künftig „International Conference on Flood Management (ICFM)“ lauten soll (ISFD4, 2008). Letztlich trägt dieser begriffliche Wandel lediglich dem inhaltlichen Wandel Rechnung, welcher seit Jahren gefordert wird. Das Thema Hochwasserschutz beziehungsweise Hochwassermanagement muss in allen Fällen

integral betrachtet werden. Das heißt, Hochwassermanagement ist eine Querschnittsaufgabe, die nicht sektoral bewältigt werden kann. Folgerichtig wird in dem oben genannten Abschlusspapier erklärt, dass Hochwassermanagement definiert ist als das Management des Hochwasserrisikos durch integrierte Maßnahmen der Gesetzgebung, der Wirtschaft, der Regierung, der Technologie und der Bildung (ebd.).

Nicht nur begrifflich, sondern auch inhaltlich hat sich ein Wandel zur Lösung des Problems Hochwasserrisiko und zu den Zielen des Hochwasserschutzes vollzogen. Beispielsweise begründet Meyer-Zurwelle (1975) den Bedarf an Hochwasserschutzmaßnahmen damit, dass in den Flussniederungen für Besiedlung, Industrialisierung, Landwirtschaft und Verkehr in zunehmendem Maße hochwasserfreie Nutzungsflächen benötigt werden. So seien Hochwasserschutzmaßnahmen vor allem zum Zweck der Sicherung der Menschen und der hohen Sachwerte dieser Nutzungsgebiete erforderlich. Hier ist zu erkennen, dass seit Jahrtausenden primäres Ziel die anthropogene Nutzung bzw. Besiedlung dieser überschwemmungsgefährdeten Gebiete, vor allem unter Einsatz technischer Hochwasserschutzmaßnahmen, war. Dementsprechend schlägt (ebd.) prinzipiell zwei Möglichkeiten zum Hochwasserschutz vor, die alternativ oder gemeinsam zur Anwendung kommen können:

- Hochwasserschutz durch flussbauliche Maßnahmen (einschließlich Eindeichung, Wehrbau, Bau von Flutkanälen u. a.)
- Hochwasserschutz durch Hochwasserrückhaltebecken als Einzelsystem oder als Gruppe bestehend aus mehreren Systemelementen

Während der Hochwasserschutz der 70er Jahre vornehmlich die Nutzung hochwasserfreigelegter Flächen vorsah, wird heute unter anderem versucht, diesen in Anspruch genommenen Raum dem Fluss zurückzugeben. Wenige Jahre später formulierte Dyck (1980b), dass für den Hochwasserschutz der natürliche Gebietsrückhalt die größte Bedeutung hat, da er meist um einige Größenordnungen über dem durch Speicherbauten erzielbaren künstlichen Rückhalt liegt. Des Weiteren seien durch zweckentsprechende Landnutzung günstige Infiltrations- und Rückhaltebedingungen zu schaffen.

Anfang der 1980er-Jahre nahm das gesellschaftliche Umweltbewusstsein deutlich zu. Dies führte auch zu einem Umdenken in der Wasserwirtschaft. Fließ- und Standgewässer wurden nicht mehr nur als „Vorfluter“, sondern als Lebensraum für Tiere und Pflanzen ge-

sehen (Lüderitz und Jüpner, 2009). Während der 80er Jahre wurden auch die Retentionsvolumina in Form von Rückhaltebecken weiter ausgebaut. So wurden beispielsweise in Baden-Württemberg bis 2001 ca. 400 größere Rückhalteräume vorwiegend mit einem Stauinhalt zwischen 50.000 m³ und 2.000.000 m³ gebaut (Giesecke, 2001).

Plate (2008) fasst in seinem abschließenden Vortrag des 4th International Symposium on Flood Defence wie folgt zusammen: „Weder allein technische noch allein natürliche Maßnahmen zum Schutz vor Hochwasser sind eine Lösung des Problems, nur die Kombination beider leistet einen wirksamen Beitrag zur Reduzierung des Hochwasserrisikos" (Tabelle 2-1). Hochwasser sind nicht vermeidbar, aber in ihrer Wirkung minderbar (R. Jüpner, 2005).

Tabelle 2-1: Beispiele für differenzierte Maßnahmen zur Hochwasservorsorge bei unterschiedlichen Hochwasser-Wiederkehrintervallen T in Jahren (Kron, 2003 verändert).

Hochwasser mit hoher Wahrscheinlichkeit (T < 10 a)	Strukturelle Maßnahmen <ul style="list-style-type: none"> – Renaturierung – Verbesserte Infiltration, Entsiegelung – Dezentraler Rückhalt – Deichrückverlegung, Querschnittsaufweitung – Deiche
Hochwasser mit mittlerer Wahrscheinlichkeit (T = 10-200 a)	Technische Maßnahmen <ul style="list-style-type: none"> – Rückhaltebecken, -flächen – Deiche – Polder – Deichrückverlegung, Querschnittsaufweitung
Hochwasser mit niedriger Wahrscheinlichkeit (T > 200 a)	Organisatorische Maßnahmen <ul style="list-style-type: none"> – Notentlastungen – Katastrophenbewältigung – finanzielle Vorsorge

Mit den in dieser Arbeit entwickelten Methoden wird es möglich sein, strukturelle und technische Maßnahmen integriert zu betrachten. Es kann die Effektivität von Maßnahmen unter Berücksichtigung von häufigen und seltenen Überschwemmungen ermittelt werden und auf diese Weise den besten „Mix“ aus technischen und natürlichen Maßnahmen zu bestimmen.

2.2 Optimierung von Speichersystemen – Stand der Wissenschaft

Die folgende Tabelle gibt einen Überblick über die Optimierung von Speichern ab 1971 ohne den Anspruch auf Vollständigkeit. In der linken Seite werden jeweils das Jahr und die gewählte Methode genannt.

Tabelle 2-2: Erste Arbeiten zur Optimierung von Speichersystemen

1971 Lagrange-Multiplikatoren (LP)	Bereits Anfang der 70er Jahre wurden Algorithmen für die Optimierung von Hochwasserschutzmaßnahmen eingesetzt. Hughes (1971) benutzt erstmals ein mathematisches Optimierungsverfahren mit Lagrange-Multiplikatoren zur Ermittlung optimaler Abgabestrategien für Hochwasserrückhaltebecken. Als Zielfunktionen dienen ihm hierbei geschätzte Schadenkurven.
1975 Dynamische Programmierung (DP)	Meyer-Zurwelle (1975) optimiert ebenfalls Abgabestrategien von Hochwasserspeichersystemen, allerdings nicht mithilfe der Lagrange-Multiplikatoren, sondern durch Dynamische Programmierung (DP). Das Optimierungsproblem mit vielen Variablen wird dabei durch dynamische Dekomposition basierend auf dem Prinzip der dynamischen Optimierung in viele Teilprobleme mit wenigen Variablen zerlegt.
1976 Mixed-Integer linear Programming Modell (LPM)	Ein Mixed-Integer linear Programming Modell wird von Doyle et al., (1976) vorgeschlagen. Allerdings gelang es ihm nicht, hydrologische und hydraulische Zusammenhänge von Speichersystemen sowie realistische Kostenfunktionen in das Modell zu integrieren.
1977 Gradientenverfahren	Ostrowski (1977) optimiert die Entnahmesteuerung von Hochwasserspeichern im Verbund mittels Gradientenverfahren
1979 Branch-and-Bound Verfahren	Bogardi (1979) ermittelt die optimale Ausbaureihenfolge von Hochwasserrückhaltebecken bei gegebener Größe der Finanzmittel und sonstigen Beschränkungen nach einem oder mehreren Gütekriterien mit Branch-and-Bound Verfahren.
1980 Gradientenverfahren	Baumgartner (1980) setzt das mathematische Optimierungsverfahren der reduzierten Gradienten auf Hochwasser-Steuerungsprozesse ein und geht von schrittweisen vorhergesagten Zu- und Abflüssen oberhalb bzw. unterhalb der Speicher aus, wie es eine Echtzeitsteuerung verlangt. Im Gegensatz zu Bogardi (1979), der Nutzenfunktionen zur Bewertung der Effizienz von Hochwasserrückhaltesystemen einsetzt, verwendet Baumgartner (1980) vom Flussgebiet abhängige Schadenskurven um nicht zu verhindernde Hochwasserschäden so weit wie möglich zu reduzieren.

In vielen aktuelleren Arbeiten wird die Dynamische Programmierung zur Optimierung von Speichern eingesetzt. Die Anwendung dieser Algorithmen setzt allerdings in allen Fällen

voraus, dass das komplexe Speichersystem durch Systemdekomposition in mehrere einfachere, kleinere Teilsysteme zerlegt werden kann. Eine Stärke der dynamischen Programmierung ist, dass es keine Beschränkungen der Art der Zielfunktion und der Randbedingungen gibt. Es können also problemlos nichtlineare, diskrete oder sogar abschnittsweise definierte Funktionen verwendet werden (Brass, 2006). Der wesentliche Nachteil der DP ist, dass sie unter dem „Fluch der Dimensionalität“ leidet (Brass, 2006; Jairaj und Vedula, 2000). Die grundlegende Vorgehensweise der DP ist die Berechnung aller Erfolg versprechenden Möglichkeiten, um von einem Ausgangszustand in einen Endzustand zu gelangen. Dazu ist in gewissem Umfang eine Enumeration der möglichen diskreten Zwischenschritte erforderlich. Nimmt also die Anzahl der diskreten Zustände zu, so erhöht sich der erforderliche Rechenaufwand (ebd.). Jairaj und Vedula (2000) umgehen dieses Problem, indem sie eine Fuzzy Linear Programming Model verwenden. Zwar entfällt beim Fuzzy Linear Programming der „Fluch der Dimensionalität“, dennoch muss das Optimierungsmodell in der Fuzzy Programmierung abgebildet werden. Tabelle 2-3 stellt einige Arbeiten zur Dynamischen Programmierung vor.

Tabelle 2-3: Speicheroptimierung mittels Dynamischer Programmierung

1987 Dynamische Programmierung (DP)	Ormsbee et al. (1987) integrieren Aspekte der Wasserqualität in Rückhalte-systeme und erweitern damit die Anwendung der Dynamischen Programmierung (DP) auf zwei verschiedene Zielsetzungen.
1989 Stochastisch Dynamische Programmierung (SDP)	Kularathna und Bogardi (1989, 1990) verwenden Aggregations- und Dis-aggregationsverfahren zur Optimierung des Mahaweli-Talsperrensystems in Sri Lanka. Dabei fassen sie geeignete Speicher zusammen oder vernachlässigen diese, falls sie eine untergeordnete Rolle spielen, um das System mittels Stochastisch Dynamischer Programmierung (SDP) optimieren zu können. Nach der Optimierung muss das vereinfachte System wieder disaggregiert werden, um die ursprünglichen Speicher zu erhalten.
1998 Dynamische Programmierung (DP)	Turgeon und Charbonneau (1998) (zitiert nach: Brass (2006)) fassen ein Speichersystem mit insgesamt 26 Talsperren in zwei Schritten zu einem Speicher zusammen. Nach der Optimierung des Gesamtsystems erfolgte die Zerlegung in die verschiedenen parallelen Flussläufe durch Dynamische Programmierung (DP). Diese wurden jeweils in einem weiteren Schritt mittels Nicht Linearer Programmierung in die einzelnen Talsperren zerlegt.
2000 Dynamisches Modell	Olsen et al. (2000) beschreiben ein Dynamisches Modell für das Management von Überschwemmungsflächen unter Verwendung des Markov-Entscheidungs-

	<p>Modells. Dies ermöglicht einerseits die Abbildung technischer Hochwasserschutzmaßnahmen wie Deiche, Hochwasserschutzmauern, Modifizierungen am Gerinne, Rückhaltemaßnahmen und Mehrzweckspeicher, aber auch strategische Maßnahmen wie z. B. Versicherungen gegen Schäden aufgrund von Hochwasser, Restriktionen hinsichtlich der Landnutzung, Einschränkungen der Siedlungsentwicklung, Umsiedlungen, Organisation und Verhaltensvorsorge, Hochwasserwarndienste und Evakuierungspläne.</p>
<p>2000 Diskrete Differenziale Dynamische Programmierung (DDDP)</p>	<p>Sharif und Wardlaw (2000) optimieren Energiegewinnung und Bewässerung eines Mehrspeichersystems mittels genetischer Algorithmen und vergleichen die Ergebnisse mit der Diskreten Differenzialen Dynamischen Programmierung (DDDP). Sie stellen fest, dass die Ergebnisse des genetischen Algorithmus nur geringfügig von den Ergebnissen der DDDP abweichen. Im Vergleich zur Dynamischen Programmierung sei es vorteiliger, dass der Entscheidungsraum nicht diskretisiert werden muss und keine Trajektorien zur Initiierung ermittelt werden müssen. Sie stellen aber auch fest, dass bei sehr vielen Parametern die Wahrscheinlichkeit das Optimum zu finden, sinkt. Des Weiteren weisen sie darauf hin, dass das Ergebnis sehr sensitiv auf die Mutationswahrscheinlichkeit reagiert und diese daher der Problemstellung angemessen gewählt werden muss.</p>
<p>2000 Stochastische Dynamische Programmierung (SDP)</p>	<p>Jairaj und Vedula (2000) verwenden Fuzzy Linear Programming (FLP, eine Verbindung aus Fuzzy Logik und Linearer Programmierung), um bei einem Mehrspeichersystem die Abweichung zwischen Entnahme und Bedarf für die Bewässerung zu minimieren. Dabei war es insbesondere das Ziel eine Alternative zur klassischen Stochastischen Dynamischen Programmierung (SDP) zu entwickeln, da diese bei einer großen Anzahl von Becken nur schwerlich anzuwenden sei. Die Ergebnisse zeigen, dass Fuzzy Linear Programming eine gute Alternative bei der Optimierung von Mehrspeichersystemen sind.</p>
<p>2005 Genetischer Lineare Programmierung (LP)</p>	<p>Reis et al. (2005) verfolgen einen hybriden Ansatz. Sie verwenden genetische Algorithmen als auch Lineare Programmierung zur Optimierung der Entscheidungen im operationellen Betrieb von Speichern. Bei diesem Ansatz wird die Zahl der operationellen Parameter reduziert und mit dem Genetischen Algorithmus (GA) und der Linearen Programmierung (LP) optimiert. Er wendet dieses Verfahren auf ein aus der Literatur bekanntes hypothetisches System an und kommt zu dem Schluss, dass dieses Verfahren ebenso gute Lösungen liefert wie das Stochastic Dual Dynamic Programming (SDDP, eine Variation der SDP) Verfahren.</p>
<p>2006 Stochastisch Dynamischer Programmierung</p>	<p>Brass (2006) optimiert ebenfalls den Betrieb von Talsperren allerdings mittels Stochastisch Dynamischer Programmierung (SDP). Er entwickelt ein System, welches den Anwender bei der Ermittlung der optimalen Betriebsweise eines Mehrziel- und Mehrspeichersystems unterstützt. Seine Optimierung beruht auf</p>

(SDP)	expliziter Stochastisch Dynamischer Programmierung (SDP) und berücksichtigt die stochastischen Eigenschaften der Talsperrenzuflüsse. Dabei aggregiert er unterschiedliche Zielgrößen mittels einheitenloser Strafpunkte und Zielgewichte. Durch eine automatische Variation dieser Zielgewichte im Gesamtsystem ermöglicht er eine interaktive a posteriori Auswahl.
2006 Heuristischen Dekompositions- Aggregationsansatz	Winkler (2006) optimiert Hochwasserrückhaltebecken und klassifiziert das Optimierungsverfahren als heuristischen Dekompositions-Aggregationsansatz mit iterativer Teilsystemoptimierung. Hierbei wurde Wert auf ein möglichst einfaches und transparentes Optimierungsverfahren gelegt. Damit die Rechenzeiten des Optimierungsmodells dem Echtzeiteinsatz genügen, werden suboptimale Lösungen in Kauf genommen und als optimierte Lösungen bezeichnet.

Während in der Mathematik evolutionäre Algorithmen zur Lösung eines Optimierungsproblems selten eingesetzt werden, da sie weder das Auffinden des Optimums noch eine Mindestqualität der Lösung garantieren können, sind sie im Ingenieurwesen häufig verwendete Werkzeuge. Sie sind robust, hinreichend effektiv und flexibel genug, um sie bei einer breiten Palette von Problemstellungen einzusetzen. Bedingt durch den probabilistischen Ansatz dieser Verfahren liegt der wesentliche Vorteil darin, dass nur wenige und einfach strukturierte Informationen über die zu lösenden Problemstellungen benötigt werden. Daher ist es möglich detaillierte und bewährte Simulationsmodelle zur Evaluierung der Qualitätswerte einzusetzen. Dabei können die Zielfunktionen mono- oder multikriteriell und beliebig komplex sein und sind ohne Schwierigkeiten in das Optimierungssystem integrierbar. Tabelle 2-4 stellt einige aktuelle Arbeiten vor.

Tabelle 2-4: Speicheroptimierung mittels Evolutionärer Algorithmen

1995 Genetischer Algorithmus (GA)	Otero et al. (1995) setzen genetische Algorithmen und dynamische Programmierung für die Planung von Rückhaltebecken im South Florida Water Management District ein. Hierbei wurde der Genetische Algorithmus (GA) erfolgreich mit einem Hochwasserwellenablaufmodell verbunden. Im Gegensatz zu den zuvor genannten Arbeiten konnten hier Modell und Optimierungsalgorithmus unabhängig betrieben werden. Bei dieser Methode wird die Verbindung zwischen Modell und Optimierungsalgorithmus auf den Austausch von zu optimierenden Parametern und Ergebnissen der Modellrechnung reduziert.
2001 Evolutionstrategie (ES)	Lohr (2001) entwickelt ein konzeptionelles Modell zum Betrieb und zur Bewertung wasserwirtschaftlicher Speichersysteme. Für die Optimierung der Betriebsregeln setzt er einen evolutionstrategischen Algorithmus ein, allerdings ist dieser noch auf die Optimierung eines Ziels und kontinuierlicher Parameter

	ausgelegt.
2002 / 2003 Evolutionstrategie (ES)	Mehler und Leichtfuß (2001, 2003) optimieren ebenfalls mit Evolutionsstrategie und Einzelzielsetzung Stadtentwässerungsnetze.
2006 Evolutionstrategie (ES)	Muschalla (2006) entwickelt einen evolutionären Algorithmus zur multi-kriteriellen Optimierung von Stadtentwässerungsnetzen und ermöglicht durch Integration des Paretodominanzkriteriums die multikriterielle Optimierung.
2009 Evolutionstrategie (ES)	Feldmann (2009) optimiert den Mitteleinsatz für Hochwasserschutzmaßnahmen an Fließgewässern im urbanen Raum mittel evolutionstrategischer Algorithmen.

2.3 Forschungsbedarf

Bei obigen Listen mit aufgeführten Arbeiten wird zwar nicht der Anspruch auf Vollständigkeit erhoben, dennoch fällt auf, dass keine der Arbeiten eine kombinatorische Optimierung erlaubt. Die Möglichkeit der kombinatorischen Optimierung ist aber zwingend erforderlich, um Maßnahmen hinsichtlich ihrer Lage im Raum zu optimieren. Des Weiteren wurde bei Recherchen deutlich, dass es keine Algorithmen gibt, die reelle als auch nominal diskrete Parameter optimieren und die auf die hier vorliegende ingenieurtechnische Problemstellung der Hochwasserspeicheroptimierung übertragbar ist. Allerdings ist die Berücksichtigung der Volumina und Abgaben bei der Dimensionierung von Hochwasserspeichern ein wesentlicher Punkt. Daher ist es erforderlich, eine neue Generation von Optimierungsalgorithmen zu entwickeln, die den hier gestellten Anforderungen gewachsen sind und die komplexe Problemstellungen lösen können.

Die Innovation dieser Arbeit liegt in dem neu entwickelten evolutionären Algorithmus für nominell diskrete Repräsentanzen, sowie in dem darauf aufbauenden Memetischen Evolutionären Algorithmus. Aus ingenieurtechnischer Sicht ist damit sowohl die rein kombinatorische (ganzzahlige) als auch die memetische (gemischt-ganzzahlige) Optimierung von allen modellseitig abbildbaren Maßnahmen, genauer gesagt Speichern, möglich. Auch mit einem reellwertigen Algorithmus ist es möglich, nach guten Maßnahmenkombinationen zu suchen. Dafür sind aber mehrere Optimierungsläufe erforderlich, wobei nach jedem Optimierungslauf überprüft werden muss, welche Maßnahmen vom Algorithmus nicht oder kaum aktiviert werden. Diese müssen schrittweise zu Null gesetzt werden. Außerdem ist es mit dieser Methode nicht möglich alternative Maßnahmen

an demselben Standort zu platzieren. Ebenso muss auf eine multikriterielle Optimierung verzichtet werden, welche bei der Ermittlung der Paretofront keine manuellen Eingriffe erlaubt.

Aus mathematischer Sicht ist die Verknüpfungsmethode nominell diskreter und reeller Operatoren innovativ. Bedingt durch die Konzentration auf praxisnahe ingenieurtechnische Problemstellung war es erforderlich, eine eigene Prozedur und einen neuen Selektionsoperator für die memetische evolutionäre Optimierung zu entwickeln.

2.4 Grundlagen evolutionärer Algorithmen

Ein Evolutionärer Algorithmus (EA) ist ein Optimierungsverfahren, das als Vorbild die biologische Evolution hat. Die wesentlichen Prozesse sind die Mutation, die Rekombination und die Selektion. Evolutionsstrategien (ES) als auch Genetische Algorithmen (GA) gehören zur Gruppe der evolutionären Algorithmen. Während bei beiden der Selektionsprozess vergleichbar verläuft, unterscheiden sie sich stark in den Prozessen der Mutation und Rekombination. Während bei der Evolutionsstrategie diese Prozesse auf Basis des zu optimierenden Datentypes erfolgen, werden die Datentypen bei den genetischen Algorithmen zunächst in Form binärer Zahlen umgewandelt. Die Rekombinations- und Mutationsprozesse werden dann auf Basis dieser bit-codierten Repräsentanz des Genotyps durchgeführt.

2.4.1 Klassifizierung der Optimierungsalgorithmen

Eine einheitliche Klassifizierung von Optimierungsalgorithmen ist in der Literatur nicht zu finden, da viele Optimierungstechniken miteinander in Beziehung stehen und es insbesondere bei hybriden Algorithmen viele Kombinationen aus Einzelalgorithmen gibt. Meist werden Algorithmen, wie auch in dieser Arbeit, speziell zur Lösung einer Problemstellung entwickelt.

Papageorgiou (1996) differenziert zwischen statischen, dynamischen und stochastischen Verfahren, während Simonovic (1992) in die drei Hauptgruppen lineare Optimierung, nicht-lineare Optimierung und dynamische Programmierung unterteilt. Nach Wolbring (1997) sind die von Simonovic (1992) beschriebenen Hauptgruppen in deterministischer oder stochastischer Umgebung einsetzbar.

Schuetze et al. (2002) geben eine Übersicht in Form einer Baumstruktur, in der zu Beginn unter lokalen und globalen Methoden unterschieden wird und dann Untergruppierungen definiert werden. Als **lokale Optimierung** wird die Optimierungsaufgabe angesehen, von einem gegebenen Punkt im Gebirge aus das nächste relative (lokale) Minimum oder Maximum in der Nachbarschaft zu finden. Besteht die Aufgabe darin, das absolute Minimum oder Maximum im gesamten Gebirge zu finden, wird von **globaler Optimierung** gesprochen. Analog werden die Begriffe **lokales Optimum** und **globales Optimum** verwendet. Dochain und Vanrolleghem (2001) unterscheiden zwischen lokalen ableitungsbasierten Algorithmen, lokalen ableitungsfreien Verfahren und globalen Verfahren. Huisinga et al. (2009) veranschaulicht diese Unterscheidung mit den beiden Fragen „Wo ist das nächste Tal?“ und „Wo ist das tiefste Tal?“. Gegenstand der Betrachtung sei die folgende Klassifizierung in Abbildung 2-2 von Syrjakow (2005):

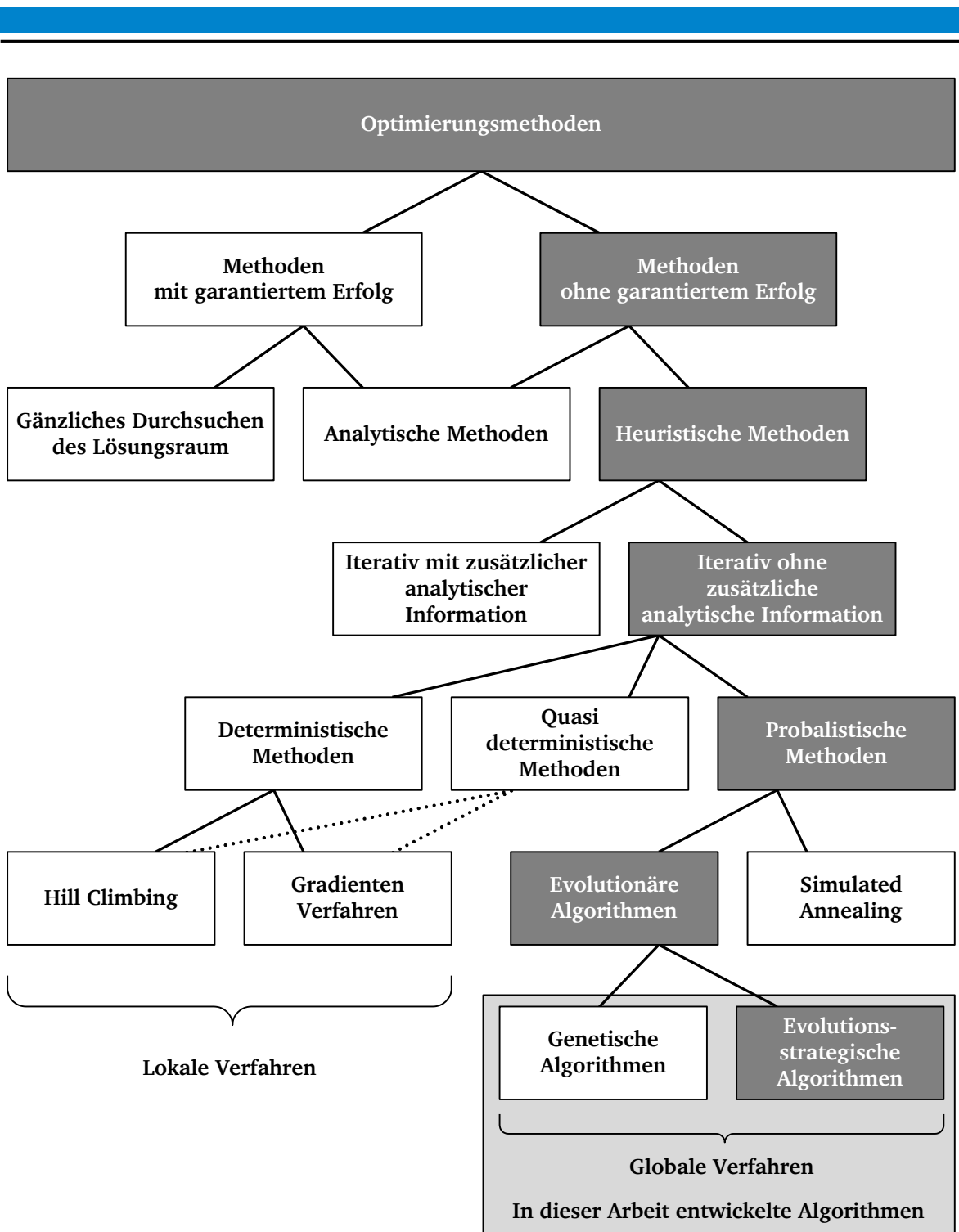


Abbildung 2-2: Klassifizierung der Optimierungsmethoden angelehnt an Syrjakow (2005)

Syrjakow (2005) differenziert zunächst zwischen Methoden mit und ohne garantiertem Erfolg. In der zweiten Ebene stehen die Methoden, die alle theoretisch möglichen Lösungen berechnen, den analytischen und heuristischen Verfahren gegenüber. In der Mitte stehen analytische Methoden, welche die optimalen Werte durch Differenziation der

Gleichung und Nullsetzen ihrer ersten Ableitung sehr schnell erreichen und unter Umständen ein Optimum garantieren können.

Die hier entwickelten Algorithmen sind eindeutig den evolutionären Algorithmen zuzuordnen. Allerdings enthalten sie sowohl Teile der genetischen Algorithmen als auch Teile evolutionsstrategischer Algorithmen. Ein wesentliches Merkmal der genetischen Algorithmen ist die Bitcodierung der Repräsentation. Diese wird hier nicht eingesetzt, was auf einen evolutionsstrategischen Algorithmus verweist. Allerdings werden für den nominell diskreten Algorithmus klassische Reproduktions- und Mutationsoperatoren der genetischen Algorithmen verwendet. Sie werden für die Hochwasserspeicheroptimierung nur unwesentlich modifiziert um sie auf die gewählte Repräsentation anwenden zu können. Für den memetischen Algorithmus werden zusätzlich die Operatoren aus den Arbeiten von Rechenberg (1973; 1994) und Schwefel (1987) eingesetzt. Diese Arbeiten stellen evolutionsstrategische Algorithmen erstmalig vor.

Die Klassifizierung nach lokaler Optimierung und globaler Optimierung dagegen ist eindeutig. Beim nominell diskreten Algorithmus handelt es sich um eine klassische globale Optimierung, beim Memetischen Algorithmus um eine globale Optimierung mit global lokaler Suche. Kapitel 3.2 (Seite 59) geht näher auf das Thema „lokal“ und „global“ ein, Kapitel 7.3 zeigt den globalen und lokalen Lösungsraum am Beispiel der Hochwasserspeicheroptimierung an der Erft.

2.4.2 Historische Entwicklung evolutionärer Algorithmen

Erste Computerprogramme, welche die Evolution simulierten, wurden Mitte der Fünfziger Jahre von Barricelli (1954, 1957) entwickelt. Er wurde dabei von John von Neumann unterstützt, der nach dem erfolgreichen Manhattan-Projekt in Los Alamos (Bau der Atombombe) entschied, dass es an der Zeit sei, einen digitalen Computer zu bauen. Er erklärte: „Ich denke jetzt über wesentlich wichtigere Dinge nach als Bomben. Ich denke dabei an Computer.“ Darüber hinaus dachte er über selbst „reproduzierende Automaten“ und den Sinn des Lebens nach (Dyson, 2009).

Ob der erfolgreichen Entwicklung dieser ersten künstlichen Evolution von Barricelli und Neumann wurden diese Optimierungsmethoden erst durch die Arbeiten von Rechenberg (1973) und Schwefel (1974) bekannt. In den 1960er und Anfang der 1970er konnte die

Gruppe um Schwefel komplexe ingenieurtechnische Probleme mit Hilfe von Evolutionsstrategien (ES) lösen (Rechenberg, 1973; Schwefel, 1974, 1977 und 1981). Ein weiterer Ansatz war die Evolutionäre Programmierung (EP) von Fogel (1966), die für die Entwicklung künstlicher Intelligenz konzipiert war. Anfang der Siebziger Jahre wurden durch Holland (1975) genetische Algorithmen (GA) entwickelt, die den Genotyp in Form eines Bitstrings repräsentieren. Die drei Ansätze entwickelten sich etwa 15 Jahre lang separat. Der wesentliche Unterschied zwischen Hollands und Rechenbergs Ansatz bestand darin, dass Holland reellwertige Zahlen durch sog. Bitstrings (das binäre Pendant zur reellwertigen Zahl) ausdrückte. Die Repräsentation der Parameter in einer binären Zahl erforderte andere Reproduktions- und Mutationsoperatoren. In den Neunzigern wurden Evolutionsstrategien und genetische Algorithmen unter dem Begriff „evolutionary computing“ zusammengefasst, da es zwar verschiedene „Dialekte“ waren, aber letzten Endes die gleiche Technologie dahinter steckte. Heute verschwimmen die Grenzen stark. Gwiazda (2006; 2007) listet hunderte von Operatoren unter dem Titel „Genetic Algorithm Reference“ auf. Dabei kennzeichnet er die Operatoren mit ihrem Anwendungsbereich.

- Binär codierte Problemstellungen
- Reell codierte Problemstellungen
- Integer codierte Problemstellungen

Die meisten der aufgelisteten Operatoren sind allerdings für zwei oder mehr Codierungsarten geeignet.

Grundprinzip dieser Algorithmen ist, dass sich Individuen bzgl. der Selektionsbedingungen als möglichst geeignet behaupten müssen und dementsprechend ihre Eigenschaften vererben dürfen oder nicht. Im Laufe mehrerer Generationen entwickelt sich so eine Menge von Individuen, die immer näher am Optimum liegt. Der prinzipielle Ablauf ist in Abbildung 2-3 dargestellt.

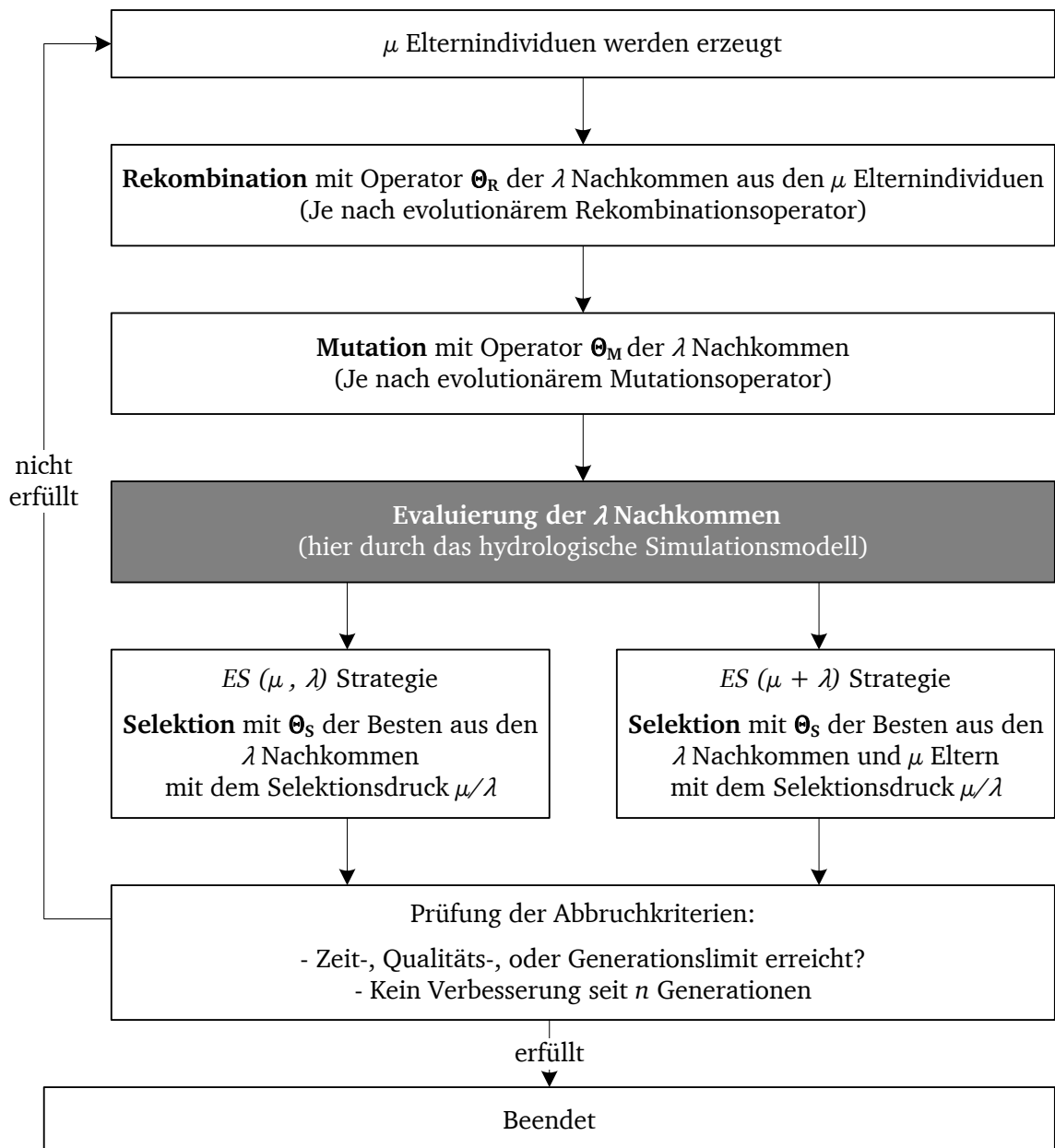


Abbildung 2-3: Prinzipieller Ablauf evolutionstrategischer Algorithmen

Auf eine ähnliche Weise wie die Natur entschieden hat, ihr Optimierungsproblem anhand des genetischen Codes zu lösen, verfahren evolutionäre Algorithmen. Hierfür wurden die biologischen Prozesse der Reproduktion, Mutation und Selektion mathematisch nachgebildet. Bei der Reproduktion Θ_R wird ein Nachfahre auf Basis zweier Eltern erstellt. Zusätzlich werden dessen Gene mit dem Operator Θ_M mutiert. Für die Selektion Θ_S werden innerhalb der Evolutionsstrategien zwei verschiedene Typen verwendet: die Komma-Strategie und die Plus-Strategie.

Bei der Komma Strategie, in der Notation nach Schwefel (1974) mit $ES(\mu, \lambda)$ notiert, werden μ Eltern selektiert, um λ Nachfahren zu generieren. Die stärksten der λ Nachfahren werden zur neuen Generation der μ Eltern. Alle anderen Nachfahren und die ursprünglichen Eltern werden verworfen.

Bei der Plus-Strategie notiert mit $ES(\mu + \lambda)$ werden die λ Nachfahren von μ Eltern generiert. Allerdings werden bei dieser Strategie die ursprünglichen Eltern bei der Selektion der stärksten λ Nachfahren weiterhin berücksichtigt. Es werden die besten λ Nachfahren, aus den $\mu + \lambda$ Individuen ermittelt, alle anderen werden verworfen. Das Wesentliche dieser Strategie ist es, dass Eltern aufgrund ihrer guten Eigenschaften durchaus über mehrere Generationen bestehen können (siehe obige Abbildung 2-3).

Die Rate μ/λ gibt den Selektionsdruck an, welche zwischen $1/3$ und $1/5$ liegen sollte (Rechenberg, 1973). Ein großer Selektionsdruck ($S < 1/5$) erhöht die Optimierungsgeschwindigkeit. Entsprechend kann nur das lokale Optimum gefunden werden. Ein kleiner Selektionsdruck ($S > 1/3$) dagegen fördert die globale Optimierung. Wird μ und λ zu 1 gesetzt entspricht dies einem einfachen Gradientenverfahren mit einem Tastschritt.

Eine mathematische Analyse der Evolutionsstrategie ist schwierig und einen mathematischen Beweis dafür, dass das Optimum gefunden wird, gibt es nicht (Schwefel, 1987 und 1994).

2.4.3 Optimierungsprobleme

In der hydrologischen Modellierung wird eine Vielzahl verschiedener Datentypen verwendet, um die hydrologischen Prozesse abzubilden. Diese reichen von den Elementardatentypen wie Festkommazahlen oder ganzen Zahlen bis hin zu Zeitreihenformaten und ihrer verschiedenen Interpretationen. Um diese Phänotypen auszuwerten, müssen Optimierungsalgorithmen auch mit verschiedenen Datentypen umgehen können. Der Begriff **Phänotyp** stammt aus der Genetik und ist definiert als die Gesamtheit der Eigenschaften eines Individuums, als Ergebnis des Zusammenwirkens von Genen.

Im konkreten Fall sind dies die mit der Modellsimulation ermittelten Evaluierungsergebnisse bzw. die Qualitätswerte eines Individuums. Der **Genotyp** ist definiert als die Summe der in den Genen lokalisierten genetischen Informationen eines Individuums. Das **Individuum** besteht aus dem Phänotyp, dem Genotyp und allen weiteren erforderlichen

Informationen für die Simulations- und Optimierungsprozesse. In der evolutionären Optimierung sind dies die Parameter, die es zu optimieren gilt.

Wesentliches Merkmal der evolutionstheoretischen Optimierung sind auch die Prozesse der **Selektion**, **Reproduktion** und **Mutation**. Diese werden auf den Genotyp angewandt. Allgemein unterscheidet man bei Datentypen zwischen reellen Variablen, ordinal diskreten Variablen und nominal diskreten Variablen. Aus diesen kann der Genotyp eines Modells bestehen.

Für die Optimierung sind je nach zu optimierendem Variablentyp geeignete Prozesse, geeignete Reproduktionsoperatoren und geeignete Mutationsoperatoren erforderlich, da nicht alle Operatoren für alle Arten der Codierung einer Problemstellungen geeignet sind. Daher muss zunächst bestimmt werden, welche Variablentypen beispielsweise in Form von Parametern bei der hier vorliegenden Problemstellung der Hochwasserspeicheroptimierung vorliegen. Bei den in dieser Arbeit betrachteten Maßnahmen handelt es sich im Wesentlichen um Hochwasserspeicher in Form von Hochwasserrückhaltebecken und Poldern. Die Operatoren der Algorithmen müssen dafür geeignet sein, die Parameter dieser Hochwasserspeicher verarbeiten zu können.

Variablentypen besitzen folgende Eigenschaften hinsichtlich ihrer Mutationsfähigkeit bzw. Veränderbarkeit:

- **Kontinuierliche Variablen r** : Diese Variablen können in beliebig kleinen Schritten verändert werden. Beispiele dieser Art Variablen sind Bodenkennwerte, Retentionskonstanten, Aufteilungsfaktoren, Abgaben etc.
- **Ordinal diskrete Variablen z** : Diese Variablen können schrittweise verändert werden und besitzen eine Rangfolge. Allerdings ist die Schrittweite vorgegeben, kann aber variabel sein. Beispiele sind Rohrdurchmesser oder Drosseinstellungen.
- **Nominal diskrete Variablen d** : Diese Variablen sind im Gegensatz zu den oberen Variablentypen in keine Rangfolge mehr zu bringen. Ein allgemeines Beispiel dieser Variablen sind Namen oder in dem hier vorliegenden Kontext der Typ (Polder oder HRB im Hauptschluss) der Maßnahme

Ziel der Optimierung ist es, einen Vektor aus n Entscheidungsvariablen zu finden, der die für die Hochwasserspeicheroptimierung erforderlichen Variablentypen enthalten darf und dessen Zielfunktionswerte eine hinreichende Güte zur Lösung des Problems besitzen. Dieser Vektor muss im Falle eines monokriteriellen Problems genau ein Ziel und im Falle eines multikriteriellen Problems k Ziele verfolgen. Die Evaluierungsfunktion $\phi: \Omega \rightarrow \Lambda$ bestimmt aus dem Entscheidungsvektor $\vec{x} = (x_1, \dots, x_n)$ des Entscheidungsraums Ω das Ergebnis y oder den multikriteriellen Ergebnisvektor $\vec{y} = (y_1, \dots, y_n)$ des Lösungsraumes $\Lambda \subseteq \mathbb{R}^k$ (Van Veldhuizen, 1999; Veldhuizen und Lamont, 2000, zitiert nach: Muschalla (2006)).

Enthält der Entscheidungsraum Ω die drei oben genannten Variablentypen, ist das Optimierungsproblem für ein monokriterielles Problem wie folgt zu definieren:

$$f(r_1, \dots, r_{n_r}, z_1, \dots, z_{n_z}, d_1, \dots, d_{n_d}) \rightarrow \min \quad (2-1)$$

$$\text{bzw.: } f(\vec{r}, \vec{z}, \vec{d}) \rightarrow \min \quad (2-2)$$

$$\text{Mit: } r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r$$

$$z_i \in [z_i^{\min}, z_i^{\max}] \subset \mathbb{Z}, i = 1, \dots, n_z$$

$$d_i \in D_i = \{d_{i,1}, \dots, d_{i,|D_i|}\}, i = 1, \dots, n_d$$

Es ist allgemein üblich von Minimierungsproblemen zu sprechen, da $\max\{F(x)\} = -\min\{-F(x)\}$ gilt. Für eine multikriterielle Problemstellung gilt entsprechend:

$$F(\vec{r}, \vec{z}, \vec{d}) = f_1(\vec{r}, \vec{z}, \vec{d}), \dots, f_k(\vec{r}, \vec{z}, \vec{d}) \rightarrow \min \quad (2-3)$$

$$\text{Mit: } k \in \mathbb{N} \text{ und } \vec{r}, \vec{z}, \vec{d} \in \Omega$$

2.4.4 Mono- und multikriterielle Optimierung

Bei der monokriteriellen Optimierung werden nach erfolgreicher Evaluierung die Individuen ihrer Qualität entsprechend sortiert. Im Vergleich zur multikriteriellen

Optimierung ist dieser Schritt weniger komplex, da die Qualitätswerte „lediglich“ aufsteigend oder absteigend geordnet werden müssen (siehe Abbildung 2-4). Liegt eine multi-kriterielle Problemstellung vor, muss für die Optimierung ein Gewichtungsvektor angegeben werden, der das multi-kriterielle Problem auf ein monokriterielles reduziert. Ein Entscheider muss also seine Präferenzen angeben, bevor die Gesamtheit der Lösungen vorliegt. Man spricht auch von der „A priori preference articulation“.

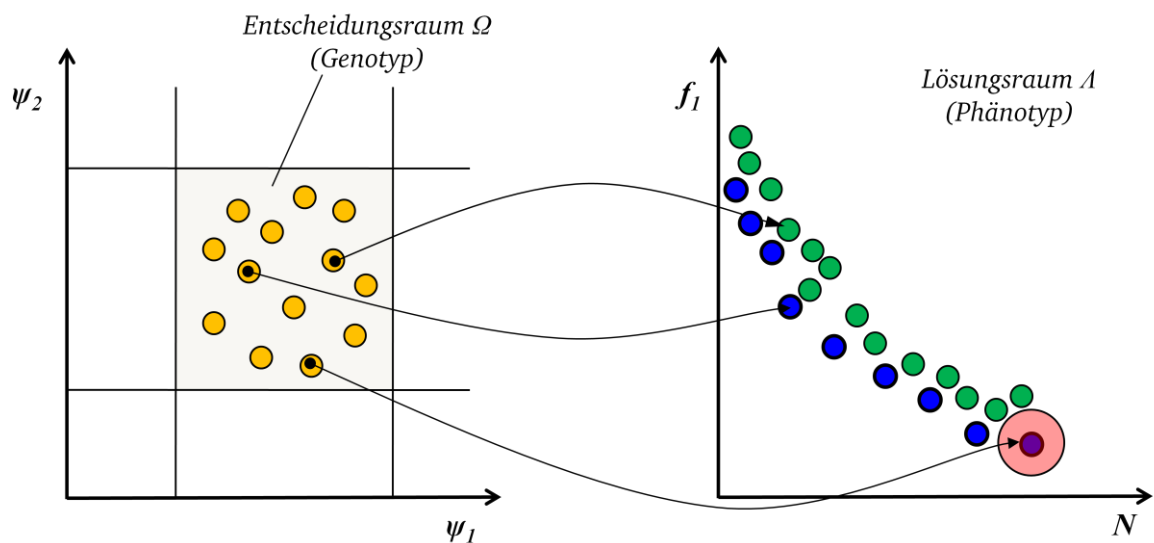


Abbildung 2-4: Die Qualitäten der Genotypen des Entscheidungsraumes Ω werden durch Simulation ermittelt und im Lösungsraum abgebildet. Dort ist f_1 der Qualitätswert und N die Zahl der Evaluierungen (Simulation und Auswertung der Zielfunktion). In Blau sind die aktuell besten Lösungen dargestellt.

Alternativ gibt es einige monokriterielle Verfahren, die dennoch mehrere berechnete Lösungen anbieten. Eine einfache Methode ist beispielsweise, die monokriterielle Optimierung mit mehreren Gewichtungsvektoren (z. B. 10 % zu 90 % oder 20 % zu 80 %). Für die multi-kriterielle Optimierung sind weitere Mechanismen erforderlich.

Da für ein multi-kriterielles Optimierungsproblem keine eindeutig „beste“ Lösung definiert ist, bestimmt man eine Menge von Lösungen des Optimierungsproblems, bei der eine Verbesserung eines Zielfunktionswertes nur noch durch Verschlechterung eines anderen erreicht werden kann, also die Menge optimaler Kompromisse. Diese Lösungsmenge bezeichnet man als *Paretomenge* oder *Paretooptimum* des zugrunde liegenden Paretooptimierungsproblems. Deren Elemente werden auch als *paretooptimal* bezeichnet.

Beispielsweise besteht der Entscheidungsraum in Abbildung 2-5 aus einem 2-dimensionalen Parameterraum und der Lösungsraum besitzt 2 konkurrierende Ziele. Der Optimierungsalgorithmus identifiziert die paretooptimale Menge mithilfe des sogenannten nicht-dominanz Kriteriums.

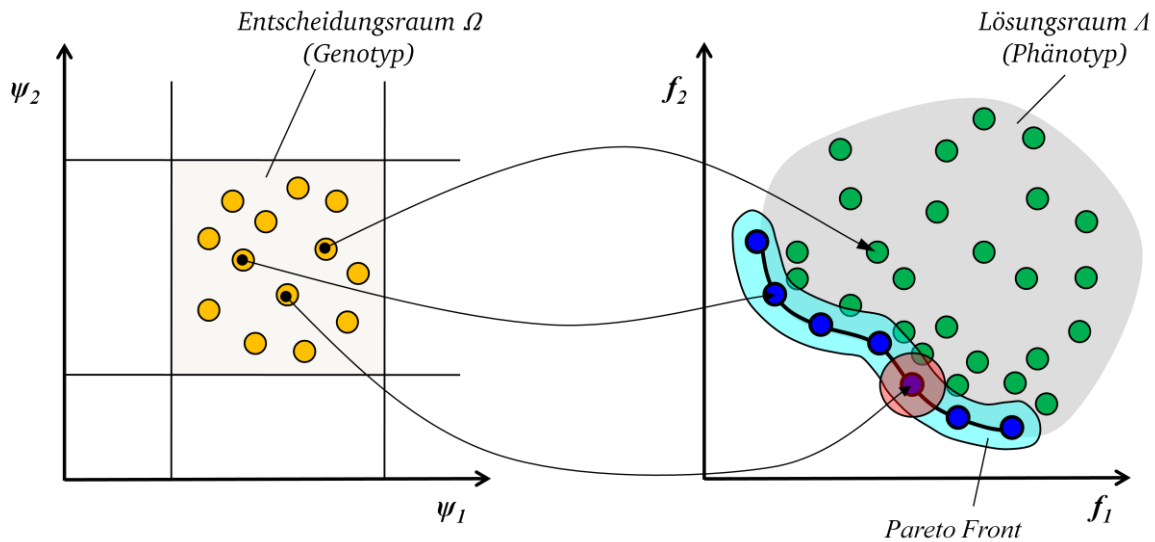


Abbildung 2-5: In Blau dargestellt die Lösungen, welche von keiner anderen Lösung dominiert werden.

Ein Lösungsvektor $\vec{u} = \{u_1, \dots, u_i\}$ ist genau dann pareto dominant bzw. nicht dominiert, wenn er immer partiell kleiner gleich und mindestens einmal partiell kleiner als der Lösungsvektor $\vec{v} = \{v_1, \dots, v_i\}$ ist. Weitere und detailliertere Informationen zu den Begriffen Pareto Dominanz, Paretooptimalität und die Menge der paretooptimalen Lösungen sind in Veldhuizen und Lamont (2000), Meyer (2002) und Zitzler et al. (2002) zu finden. Das Prinzip der Pareto Dominanz ist auch in der später folgenden Abbildung 2-7 „Beispiel einer multikriteriellen Parameteroptimierung durch die reellwertige Evolutionsstrategie“ zu erkennen. Dort wird sichtbar, dass die Paretofront bedingt durch ihre Definition Sprünge enthalten kann, da die vorderste Linie guter Ergebnisse von einer konkaven in eine konvexe Form übergeht. Die Kenntnis derartiger Sprünge kann bei Entscheidungen eine relevante Information sein. Ist beispielsweise bei der Planung eines neuen Hochwasser-rückhaltebeckens zu entscheiden ob dies am Standort A oder B realisiert werden soll, so ist der Lösungsraum dieser Fragestellung durch zwei Punkte im Raum gekennzeichnet zwischen denen keine weiteren Lösungen existieren. Soll hingegen entschieden werden wie groß die maximalen Abgaben im Hochwasserfall sein dürfen handelt es sich bei dieser

Fragestellung um einen kontinuierlichen Lösungsraum, da die Abgabe in beliebig kleinen Schritten verändert werden kann.

Ein Entscheider kann nun mit diesen detaillierten Informationen entsprechend seiner Präferenzen eine Lösung wählen. Zusätzlich erfordert eine Änderung der Präferenz keine weiteren Optimierungen, da der gesamte Lösungsraum bekannt ist. Jede dieser Lösungen ist ein „bester“ Kompromiss innerhalb des Lösungsraumes aus zwei Zielfunktionen.

2.5 Problem­domäne Hochwasserspeichersystem

In der Mathematik unterscheidet man zwischen diskreten und kontinuierlichen, kombinatorischen und relationalen, logischen und quantitativen, statischen und taktischen (spieltheoretischen) Entscheidungsproblemen (Hättenschwiler und Gachet, 2009). Dabei gibt es immer zwei Aufgaben zu lösen:

- Die Darstellung (Repräsentation) der zulässigen Entscheidungsalternativen oder eben des Entscheidungsraums und
- die Bewertung der Alternativen und die Suche nach guten bzw. nach der besten Alternative im dargestellten Entscheidungsraum.

Genauer gesagt bedeutet dies, dass für jedes Problem, das mittels evolutionärer Algorithmen gelöst werden soll, einige Punkte geklärt werden müssen, um einen geeigneten Algorithmus zu entwickeln (Blom, 2003; Gerdes u. a., 2004).

1. **Die Wahl einer problemadäquaten Repräsentation.** Die Wahl der Repräsentation hängt stark von der vorliegenden Problem­domäne ab. Sie muss zum einen die Variablen enthalten, die erforderlich sind, um die Problem­domäne hinreichend abzubilden (Genotyp). Zum anderen muss die Repräsentation eine Form besitzen, um evolutionäre Reproduktions- und Mutationsoperatoren darauf anwenden zu können.
2. **Wahl der Reproduktionsoperatoren.** Je nach Problem­domäne und Repräsentation können Reproduktionsoperatoren mit verschiedenen Eigenschaften angewendet werden, um aus Eltern Nachfahren zu kombinieren. Ziel ist es, in Abhängigkeit der Problem­domäne die vorteiligen Eigenschaften der Eltern an die Nachfahren zu vererben. Die meisten Operatoren garantieren die Validität des Nachfahren, manche aber nicht. In diesem Fall muss geprüft

werden, ob der generierte Nachfahre sowohl die Repräsentation als auch die Randbedingungen einhält.

3. **Wahl der Mutationsoperatoren.** Ebenfalls in Abhängigkeit der Problem- domäne müssen geeignete Mutationsoperatoren gewählt werden, die es er- lauben, neue Bereiche im Lösungsraum zu durchsuchen. Wie bei den Reproduktionsoperatoren muss auch hier unter Umständen die Validität der Nachfahren geprüft werden. Denn es muss garantiert werden, dass der Nach- fahre weiterhin der für eine Problem- domäne definierten Repräsentation ent- spricht.
4. **Wahl der Selektionsoperatoren.** Bei der Selektion ist zu unterscheiden zwischen der Plus-Strategie und der Komma-Strategie. Bei der Plus-Strategie gelangen Eltern plus Nachkommen in die Selektionsmenge, bei der Komma- Strategie nur die Nachkommen. Neben diesen monokriteriellen Selektions- operatoren sind im Falle von multikriteriellen Problemen multikriterielle Selektionsoperatoren erforderlich. Des weiteren kann es erforderlich sein, bei Problem- domänen bestehend aus zwei oder mehr Variablentypen spezielle Selektionsoperatoren zu entwickeln, da die verschiedenen Variablentypen zu einem inhomogenen Lösungsraum führen, was bei der Selektion berücksichtigt werden muss.
5. **Wahl der Prozedur.** Bei der Wahl der Prozedur (Ablaufschema des Algorith- mus) müssen der softwareseitige Rahmen und die Anbindung der Modelle be- rücksichtigt werden. Dies spielt in insbesondere bei einer Parallelisierung eine Rolle. Darüber hinaus sind spezielle Prozeduren erforderlich, wenn die Problem- domäne aus verschiedenen Variablentypen besteht und dafür spezialisierte Selektionsoperatoren eingesetzt werden.
6. **Wahl der Initialisierung.** In einigen Fällen ist es hilfreich, die Startpopulation mit einer anderen Methode als dem gängigen, zufälligen Setzen der Startpara- meter zu initialisieren. Die Wahl der Startpopulation kann zum einen dem Algorithmus das Finden der Lösung erleichtern. Sie kann aber auch bewusst ungünstig gewählt werden, um zu prüfen, ob der Algorithmus in der Lage ist, das Optimum zu finden.

Die oben gestellten Fragen werden in dieser Arbeit mit unterschiedlicher Gewichtung mehrfach beantwortet:

1. Im Kapitel 2.7.1 „Problem­domäne Handelsreisendenproblem (TSP)“ werden diese Fragen zunächst anhand des Problems des Handelsreisenden gelöst, um einen Basisalgorithmus zu entwickeln, der in der Lage ist, kombinatorische Probleme zu lösen.
2. Zu Beginn des Kapitels 3 „Methodik der Hochwasserspeicher­optimierung“ werden diese Punkte für die hier vorliegende Problem­domäne der Optimierung von Hochwasserspeichersystemen geklärt und der entwickelte nominell diskrete Algorithmus um entsprechende Operatoren, als auch um Schnittstellen und Funktionalitäten zur Steuerung des Simulationsmodells erweitert.
3. Im Kapitel 3.2 „Methodik der Memetischen Hochwasserspeicher­optimierung“ wird insbesondere auf die memetische Prozedur und memetische Selektion eingegangen, da die bis dahin entwickelten Operatoren wiederverwendet werden.

2.5.1 Zielfunktion

Für die hier vorliegende Problemstellung der Variantenoptimierung von Hochwasserspeichersystemen, reduziert sich Gleichung (2-2) auf

$$f(d_1, \dots, d_{n_d}) \rightarrow \min \quad (2-4)$$

Mit: $d_i \in D_i = \{d_{i,1}, \dots, d_{i,|D_i|}\}, i = 1, \dots, n_d$

da es sich zunächst um ein rein kombinatorisches Problem handelt. Ziel dieser Arbeit war es aber nicht nur die Hochwasserschutzvariante, sondern auch die zugehörigen kontinuierlichen Parameter wie z. B. das Volumen der Rückhaltebecken zu optimieren. Daher galt es einen kombinatorischen Algorithmus (Kapitel 3.1 „Methodik der nominell diskreten Hochwasserspeicher­optimierung“) zu entwickeln, der eine gute Grundlage bildet, um diesen später mit reellwertigen Operatoren zu kombinieren.

2.6 Grundlagen der reellwertigen Evolutionsstrategie

Im ersten Teil des Kapitels 3 wird zunächst ein neuer kombinatorischer evolutionsstrategischer Algorithmus für nominell diskrete Parameter entwickelt, der in Kapitel 3.2 “Methodik der Memetischen Hochwasserspeicheroptimierung“ mit den Operatoren aus den Arbeiten von Huebner (2003) aber insbesondere von Muschalla (2006) kombiniert wird, um die Memetische Optimierung zu ermöglichen.

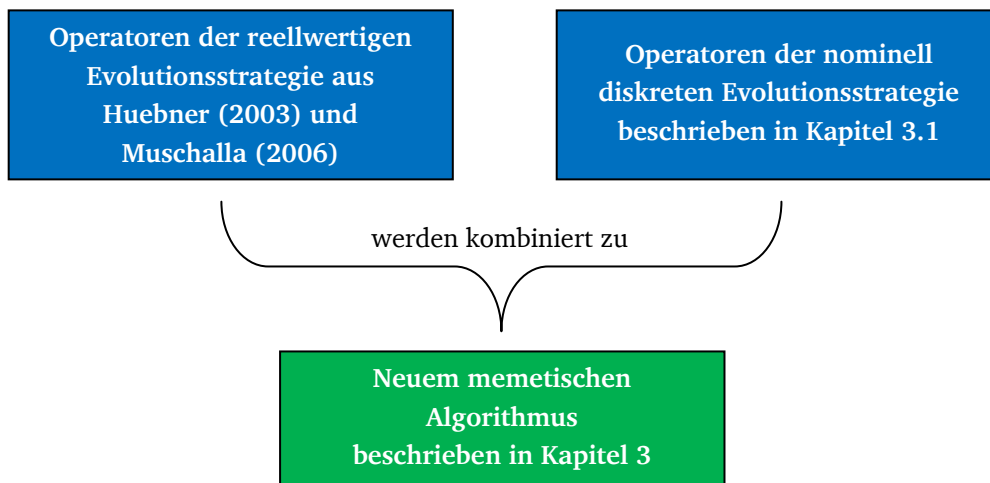


Abbildung 2-6: Bestehende und in dieser Arbeit entwickelte Algorithmen

Für den memetischen Algorithmus werden Operatoren und Repräsentationen der reellwertigen und nominell diskreten Evolutionsstrategie eingesetzt. Daher sind im Folgenden die Grundlagen des reellwertigen Algorithmus aufgeführt.

2.6.1 Repräsentation

Das Individuum dieses evolutionsstrategischen Algorithmus zur Lösung reellwertigen Probleme basiert auf einem Vektor reeller Zahlen,

$$\vec{r} \text{ mit } r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r \quad (2-5)$$

Ein Beispiel sieht folgendermaßen aus:

$$(1,3 \quad 2,5 \quad 83 \quad 17 \quad -154 \quad 0,001) \quad (2-6)$$

Zusätzlich enthält jedes Individuum entweder eine Schrittweite δ oder einen Schrittweitenvektor $\vec{\delta}$, welcher bei der Rekombination und Mutation parallel zu den eigentlichen Genen evolutioniert wird (Rechenberg, 1973; 1994; Beyer und Deb, 2000). Eine ausführliche Beschreibung des Verhaltens und der problemadäquaten Einsatzmöglichkeiten verschiedener Anpassungsverfahren ist in Muschalla (2006) zu finden.

2.6.2 Reproduktions- und Mutationsoperatoren

Obige Repräsentation erlaubt es, Problemdomänen zu optimieren, die durch reellwertige Variablen gekennzeichnet sind. Auf die Bitcodierung der genetischen Algorithmen wird hier verzichtet. Daher sind geeignete Reproduktions- und Mutationsoperatoren, wie sie von Rechenberg (1973) und Schwefel (1977) entwickelt wurden, erforderlich. Der bisherige Algorithmus zur Lösung reellwertiger Probleme enthält folgende Operatoren. Diese Operatoren werden später im Kapitel 3.2 auch beim neuen memetischen Algorithmus eingesetzt.

Tabelle 2-5: Operatoren des reellwertigen, evolutionsstrategischen Algorithmus

Reproduktionsoperatoren

- Zufälliges Ziehen eines Elter (Rechenberg, 1994; Schwefel, 1995)
 - Diskrete Rekombination nach Rechenberg (Rechenberg, 1994; Schwefel, 1995; Pohlheim, 1998)
 - Erweiterte, diskrete Rekombination (Muschalla, 2006)
 - Intermediäre Rekombination (Pohlheim, 1998)
-

Mutationsoperatoren

- Klassischer Mutationsoperator mit Schrittweitenvektor oder einzelner Schrittweite als Strategieparameter (Rechenberg, 1973 und 1994)
 - Mutationsoperator mit Schrittweitenvektor oder einzelner Schrittweite als Strategieparameter (Schwefel, 1995)
-

Selektionsoperatoren

- Kombinierte Dominanz- und Diversitätsanalyse (Van Veldhuizen, 1999; Veldhuizen und Lamont, 2000 zitiert nach Muschalla (2006))
 - Nutzung einer sekundären Population (Van Veldhuizen, 1999; Veldhuizen und Lamont, 2000)
-

2.6.3 Prozedur des reellwertigen Algorithmus

Die Prozedur des reellwertigen Algorithmus beginnt, wie im Algorithmus 2-1 dargestellt, mit dem zufälligen oder definierten initiieren der Startgeneration \vec{x} , der Schrittweite δ und auch des Schrittweitenvektors $\vec{\delta}$. Danach werden die Schleifen über die Generationen und Nachfahren ausgeführt. Abweichend zum Algorithmus 2-2, der Rundreisioptimierung, und zu dem Algorithmus 3-1 der Hochwasserspeicheroptimierung ist zu bemerken, dass hier die Prozesse zur Reproduktion, Mutation und Evaluierung sequenziell für ein Individuum innerhalb der Schleife der Nachfahren P_{ij} erfolgt und nicht für einen Individuenvektor P_i . Wurde die gesamte Population P_i evaluiert, wird diese beendet und die Eltern für die nächste Generation selektiert. P_i ist die Population der Generation i und enthält den Vektor aus Individuen. Daher entspricht P_{ij} einem Individuum.

```

1 Initialisiere:    $P_0 = \begin{Bmatrix} \vec{r}_1, \dots, \vec{r}_\mu \\ \vec{\delta}_1, \dots, \vec{\delta}_\mu \end{Bmatrix}$ 
2 Do While Not   Terminierungsbedingung
3   For i = 0 To  $\gamma - 1$ 
4     For j = 0 To  $\lambda - 1$ 
5       Reproduziere:  $P_{ij}^* \begin{Bmatrix} \vec{r}_{ij}^* \\ \vec{\delta}_{ij}^* \end{Bmatrix} = X P_{ij}$ 
6       Mutiere:       $P_{ij}^{**} = \begin{Bmatrix} \vec{r}_{ij}^{**} \\ \vec{\delta}_{ij}^{**} \end{Bmatrix} = \Theta P_{ij}^*$ 
7       Evaluiere:     $P_{ij}^{***} = \Phi \{ \vec{r}_{ij}^{**} \}$ 
8     Next j
9     Selektiere:      $P_{i+1} = \Xi(P_i^{***} \cup P_i)$    (Plus-Strategie)
10    or
11    Selektiere:      $P_{i+1} = \Xi(P_i^{***})$          (Komma-Strategie)
12    or
13    Selektiere:      $P_{i+1} = \Xi_M(P_i^{***})$        (Multikriterielle Selektion)
14  Next i
15 Loop

```

<i>mit:</i> $r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r$	<i>Reeller Entscheidungsraum/Genotyp</i>
γ	<i>Anzahl der Generation.</i>
μ	<i>Anzahl der Eltern.</i>
λ	<i>Anzahl der Nachfahren.</i>
$X(\mu)$	<i>Reproduktionsoperator</i>
Θ	<i>Mutationsoperator</i>

ϕ	<i>Evaluierungsfunktion (Simulationsmodell)</i>
\mathcal{E}	<i>Selektionsoperator</i>
\mathcal{E}_M	<i>Multikriterieller Selektionsoperator</i>
P_i	<i>Population.</i>
$\vec{r}_{ij} = r_{11}, \dots, r_{nm}$	<i>Vektor der reellen Entscheidungsvariablen</i>
$i \in \{1, 2, \dots, \mu\}, j \in \{1, 2, \dots, \gamma\}$	<i>eines Elter.</i>
$\vec{\delta}_{ij} = \delta_{11}, \dots, \delta_{nm}$	<i>Vektor der Schrittweiten eines Elter.</i>
$i \in \{1, 2, \dots, \mu\}, j \in \{1, 2, \dots, \gamma\}$	<i>Nachkommen einer Population der</i>
P_i^* bis P_i^{***} , $i \in \{1, 2, \dots, \gamma\}$	<i>unterschiedlichen Stufen.</i>
$\vec{r}_{ij}^* = r_{11}^*, \dots, r_{nm}^*$ bis	<i>Vektor der reellen Entscheidungsvariablen</i>
$\vec{r}_{ij}^{**} = r_{11}^{**}, \dots, r_{nm}^{**}$,	<i>eines Nachkommens der unterschiedlichen</i>
$i \in \{1, 2, \dots, \lambda\}, j \in \{1, 2, \dots, \gamma\}$	<i>Stufen.</i>
$\vec{\delta}_{ij}^* = \delta_{11}^*, \dots, \delta_{nm}^*$ bis	<i>Vektor der Schrittweiten eines</i>
$\vec{\delta}_{ij}^{**} = \delta_{11}^{**}, \dots, \delta_{nm}^{**}$,	<i>Nachkommens der unterschiedlichen</i>
$i \in \{1, 2, \dots, \lambda\}, j \in \{1, 2, \dots, \gamma\}$	<i>Stufen</i>

Algorithmus 2-1: Prozedur der parametrischen Evolutionsstrategie

Abbildung 2-7 zeigt das Ergebnis einer Kalibrierung mittels multikriterieller Parameteroptimierung. Die gelben Punkte stellen die ermittelten Lösungen da. In grün sind Lösungen markiert, welche zur Paretofront gehören. Eingekreist ist ein Sprung der Paretofront am dem Wechsel von einer konkaven zu einer konvexen Form.

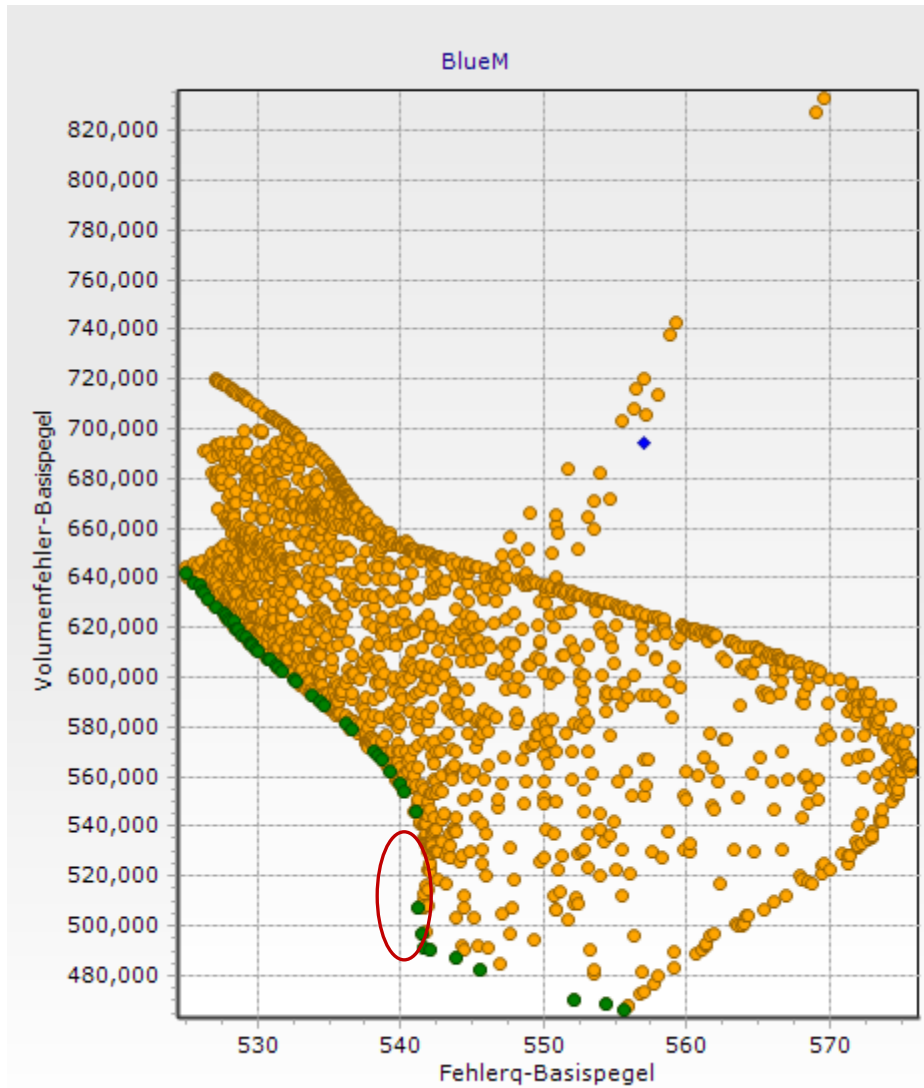


Abbildung 2-7: Beispiel einer multikriteriellen Parameteroptimierung durch die reellwertige Evolutionsstrategie. Die pereto-dominanten Lösungen sind in grün abgebildet.

Dies zeigt deutlich, dass die Parameter, die zu Lösungen in diesem Bereich geführt haben, sehr sensitiv sind. Eine kleine Variation der Parameter würde leicht dazu führen, dass deren Lösungen nicht mehr auf der Paretofront liegen. Die Optimierung begann mit den Lösungen im Bereich Fehlerquadrat = 570 und Volumenfehler = 820,000. Der Algorithmus hat sich dann in das Hauptfeld vorgearbeitet und dort versucht die Paretofront möglichst dicht zu besetzen. Dies gelang im Bereich $x \leq 540$ besser als im Bereich >540 , da die Parameter der Lösungen dort weniger sensitiv waren. Zu erkennen ist auch, dass ein Parameter offensichtlich starken Einfluss auf die Lösungen hatte. Dieser formt die obere und rechte, stark besetzte Begrenzungslinie. Weitere Untersuchungen ergaben, dass es sich um den Durchlässigkeitsbeiwert handelte.

2.7 Grundlagen der kombinatorischen Evolutionsstrategie

Die Grundlage für die nominell diskrete Optimierung bildet ein Algorithmus, der zur Lösung des kombinatorischen Problems des Handlungsreisenden (auch: Handlungsreisenden-, Rundreiseproblem oder Traveling Salesman Problem TSP) entwickelt wurde. Diese Problemstellung hat thematisch nichts mit der Problemdomäne des Hochwasserschutzes gemein. Die Entwicklung des Algorithmus wurde anhand dieses Beispiels vor allem aus praktischen Gründen vorgenommen.

Da zu Beginn weder ein geeigneter Datensatz und Schnittstellen, noch Algorithmus und Prozedur für die kombinatorische Optimierung vorlagen, musste die Problematik in Teilprobleme zerlegt werden. Daher wurde zuerst der Algorithmus entwickelt und dann ein softwareseitiger „Rahmen“. Zu dessen Erprobung und Validierung ist eine Problemstellung erforderlich, die wenig Rechenzeit benötigt, um die Prozedur und die Operatoren zu entwickeln. Die Evaluierung der Problemstellung Hochwasserschutz durch ein Simulationsmodell erfordert Sekunden bis Minuten, während die Evaluierung des Rundreiseproblems in wenigen Millisekunden erfolgt. Daher ist ein angebundenes komplexes Simulationsmodell zu Beginn der Entwicklung mehr oder weniger „hinderlich“. Des Weiteren ist es erforderlich fehlerfreie Lauffähigkeit festzustellen. Dies ist nur mit einer Problemstellung möglich, bei der das optimale Ergebnis berechnet werden kann. Diese Information ist erforderlich, um bestätigen zu können, ob das Optimum gefunden wurde.

Bei beiden Problemstellungen handelt es sich um „**kombinatorische**“ Problemstellungen. Genauer betrachtet handelt es sich aber beim Rundreiseproblem um eine **Permutation** aus ganzen Zahlen. Die Anzahl der Permutationen von n verschiedenen Elementen beträgt $n!$. Bei dem hier Problem der Hochwasserspeicher handelt es sich um eine **Variation** mit Zurücklegen und mit Beachtung der Reihenfolge, ähnlich einem Zahlenschloss. Daher gibt es n^k Möglichkeiten bei n Maßnahmen pro Standort und k Standorten. Für eine Permutation werden alle n Elemente verwendet während für eine Variation lediglich k Elemente verwendet werden. Daher sind für das Rundreiseproblem und das Hochwasserschutzproblem etwas andere Operatoren erforderlich. Die entwickelte Prozedur und der softwaretechnische Rahmen können aber weiter verwendet werden.

Auch wenn es Problemstellungen geben mag, die exakt auf die Problemdomäne Hochwasserschutz übertragbar sind, wurde das Rundreiseproblem gewählt. Es handelt sich

dabei um ein klassisches mathematisches Benchmarkproblem – viele Ergebnisse sind hinlänglich bekannt. Die erforderlichen Operatoren zur Reproduktion und Mutation der Permutation sind etwas komplexer als für die Variation. Daher ist der Schritt zum Hochwasserschutzproblem auch „einfacher“, was einen gewissen Sicherheitspuffer bietet.

Im Folgenden werden die Begriffe Permutation und Variation nur verwendet, wenn dies für das Verständnis erforderlich ist. Des Weiteren werden die übergeordneten Begriffe „Kombinatorik“ und „Kombination“ eingesetzt.

Bei den in diesem Kapitel beschriebenen Algorithmen wird die Evolutionsstrategie zur Lösung des Rundreiseproblems als kombinatorische Evolutionsstrategie oder kombinatorischer Algorithmus bezeichnet. Die Evolutionsstrategie zur Lösung des kombinatorischen Hochwasserspeicherproblems in Kapitel 3.1 wird als nominell diskrete Evolutionsstrategie oder als nominell diskreter Algorithmus bezeichnet, da zur Hochwasserspeicheroptimierung nominell diskrete Parameter eingesetzt werden. Gegebenenfalls werden die Begriffe verkürzt auf nominelle Evolutionsstrategie, nomineller Algorithmus oder nominelle Operatoren.

2.7.1 Problemdomäne Handelsreisendenproblem (TSP)

Das Problem des Handelsreisenden (TSP) ist ein kombinatorisches Optimierungsproblem, das im Wesentlichen aus den Bereichen des Operations Research und der theoretischen Informatik stammt. Es wird häufig verwendet, um kombinatorische Algorithmen zu entwickeln und zu erproben. Ziel ist es, den kürzesten Weg für einen Handelsreisenden zu finden, der beginnend in seiner Heimatstadt eine definierte Liste von Städten genau einmal besuchen muss und danach in seine Heimat zurückkehren will. Die Schwierigkeit liegt in der großen Anzahl möglicher Wege. Es gibt unterschiedliche Spezialfälle des TSP Problems. Die wesentlichen sind *symmetrische* und *asymmetrische* TSP. Beim asymmetrischen TSP können Hin- und Rückrichtung zwischen zwei Städten unterschiedlich lang sein, während beim symmetrischen die Entfernung zwischen zwei Städten immer die gleiche ist, also richtungsunabhängig. Bei einer asymmetrischen TSP ist die Strecke $A \rightarrow B$ eine andere Entfernung als die Strecke $B \rightarrow A$. Hierbei geht es eher um die mathematische Komplexität als um die Frage, warum der Fahrer sich verfährt. Die Zahl der möglichen Touren beim asymmetrischen Problem ist doppelt so groß wie beim symmetrischen Problem:

$$\begin{array}{l} \text{Asymmetrisches} \\ \text{TSP:} \end{array} \quad T = (n - 1)! \quad (2-7)$$

$$\begin{array}{l} \text{Symmetrisches} \\ \text{TSP:} \end{array} \quad T = \frac{(n - 1)!}{2} \quad (2-8)$$

Mit: T : Anzahl möglicher Touren
 n : Anzahl der Städte

Im Folgenden wird von einem symmetrischen TSP ausgegangen. Die Größe des Suchraums eines TSP hängt überexponentiell von der Anzahl der Städte ab. Beispielsweise ist die Anzahl möglicher Touren eines symmetrischen TSP mit zehn Städten 181440, während es bei zwanzig Städten bereits $6,08 \cdot 10^{16}$ mögliche Touren sind. Die Zahl der Möglichkeiten steigt also sehr schnell.

2.7.2 Repräsentation und Operatoren

Für die Optimierung dieses kombinatorischen Problems bedarf es einer Abbildungsvorschrift für die Touren des Handelsreisenden, die für die Rekombinations- und Mutationsprozesse der Evolutionsstrategie geeignet ist. Dabei muss berücksichtigt werden, dass entsprechende Reproduktions- und Mutationsoperatoren vorhanden sind, die Nachfahren in Form einer Permutation evolutionieren können.

In den Arbeiten von Larranaga et al. (1994, 1999) wird zwischen folgenden Repräsentationen (Representation) unterschieden:

- Binary Representation
- Path Representation
- Adjacency Representation
- Ordinal Representation
- Matrix Representation

Zur Abbildung des Rundreisproblems wurde die Path Representation gewählt. Die Path Representation hat sich als geeignet erwiesen, um sie auch später zur Abbildung der Hochwasserspeicher einzusetzen.

Die Path Repräsentation für das Rundreiseproblem wird definiert als der Vektor der Länge n (Anzahl der Städte) ganzer Zahlen: und hat folgende Form:

$$\vec{d} \text{ mit } d_i \in D_i = \{d_{i,1}, \dots, d_{i,|D_i|}\}, i = 1, \dots, n_d \quad (2-9)$$

Ein Beispiel sieht folgendermaßen aus:

$$T_p = (3 \ 2 \ 4 \ 1 \ 7 \ 5 \ 8 \ 6) \quad (2-10)$$

Des Weiteren gibt es viele Reproduktions- und Mutationsoperatoren, die in Verbindung mit der Path Representation angewendet werden können. Eine detaillierte Beschreibung aller Repräsentationen für das Rundreiseproblem ist der Anlage A - 1 zu entnehmen.

Zur Reproduktion und Mutation der Repräsentanz des Rundreiseproblems wurden folgende Operatoren implementiert:

Reproduktionsoperatoren:

- Partially Mapped Crossover (PMX)
- Order Crossover (OX)

Mutationsoperatoren:

- Simple Inversion Mutation (SIM)
- Translocation (3Opt)
- Translocation (nOpt, mit $n = 2$)
- Exchange Mutation Operator (EM)

Anhang Anlage A - 2 und Anhang Anlage A - 3 beschreiben die Funktionsweise und die Prinzipien der Operatoren im Detail.

2.7.3 Prozedur des Algorithmus

Die Prozedur des Algorithmus zur Rundreisoptimierung beginnt wie im Algorithmus 2-2 dargestellt mit dem zufälligen oder definierten initiieren der Startpopulation $P_i = \vec{x}$. Danach startet die Schleife über die Generationen, innerhalb der zunächst alle \vec{x}_λ Vektoren einer Generation zu \vec{x}_λ^* reproduziert und zu \vec{x}_λ^{**} mutiert werden. Nachdem alle Nachfahrenvektoren erstellt wurden, werden diese in der Schleife über alle Nachfahren mit der Evaluierungsfunktion Φ ausgewertet. Die Selektion erfolgt immer durch die *Plus-Strategie*,

sprich es werden Eltern und Nachfahren gemeinsam betrachtet, sodass Individuen über mehrere Generationen hinweg überleben können.

```

1  Initialisiere:     $P_0 = \{\vec{d}_1, \dots, \vec{d}_\mu\}$ 
2  Do While Not   Terminierungsbedingung
3      For  $i = 0$  To  $\gamma - 1$ 
4          Reproduziere:  $P_i^* = \{\vec{d}_i^*\} = X P_i$ 
5          Mutiere:      $P_i^{**} = \{\vec{d}_i^{**}\} = \theta P_i^*$ 
6          For  $j = 0$  To  $\lambda - 1$ 
7              Evaluiere:  $P_{ij}^{***} = \Phi\{\vec{d}_{ij}^{**}\}$ 
8          Next  $j$ 
9          Selektiere:  $P_{i+1} = \mathcal{E}(P_i^{***} \cup P_i)$  (Plus-Strategie)
10     Next  $i$ 
11 Loop

```

<i>mit:</i> $d_i \in D_i = \{d_{i,1}, \dots, d_{i, D_i }\}, i = 1, \dots, n_d$	<i>Nominaler Entscheidungsraum/Genotyp</i>
γ	<i>Anzahl der Generation.</i>
μ	<i>Anzahl der Eltern.</i>
λ	<i>Anzahl der Nachfahren.</i>
X	<i>Reproduktionsoperator</i>
θ	<i>Mutationsoperator</i>
Φ	<i>Evaluierungsfunktion</i> <i>(Berechnung der Strecke)</i>
\mathcal{E}	<i>Selektionsoperator</i>
P_γ	<i>Population</i>
$\vec{d}_i = d_{i1}, \dots, d_{in}, i \in \{1, 2, \dots, \mu\}$	<i>Vektor der nominalen Entscheidungsvariablen</i> <i>eines Elter.</i>
P_i^* bis $P_i^{***}, i \in \{1, 2, \dots, \gamma\}$	<i>Nachkommen einer Population</i> <i>der unterschiedlichen Stufen.</i>
$\vec{d}_{ij}^* = d_{i1}^*, \dots, d_{in}^*$ bis $\vec{d}_{ij}^{**} = d_{i1}^{**}, \dots, d_{in}^{**},$ $j \in \{1, 2, \dots, \lambda\}, i \in \{1, 2, \dots, \gamma\}$	<i>Vektor der nominalen Entscheidungs-</i> <i>variablen eines Nachkommens</i> <i>der untersch. Stufen.</i>

Algorithmus 2-2: Entwickelter Algorithmus zur Rundreiseoptimierung

2.7.4 Erprobung

Zur Erprobung und Validierung wurden zwei symmetrische (Hin- und Rückweg sind äquidistant) Testinstanzen definiert. Im Ersten wurden, wie in Abbildung 2-8 bis Abbildung 2-10 ersichtlich, die Städte in einem Koordinatensystem zufällig verteilt. In der zweiten Instanz, Abbildung 2-11 bis Abbildung 2-13, wurden die Städte in einem Kreis angeordnet.

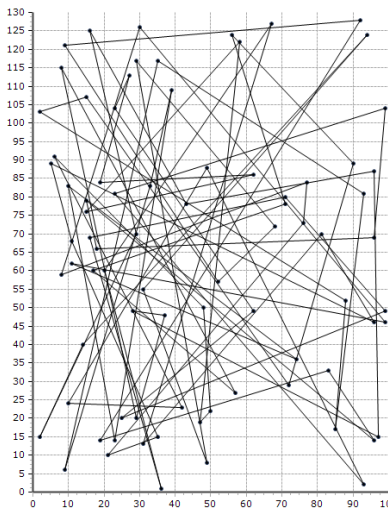


Abbildung 2-8: Zufällig, Gen. 1

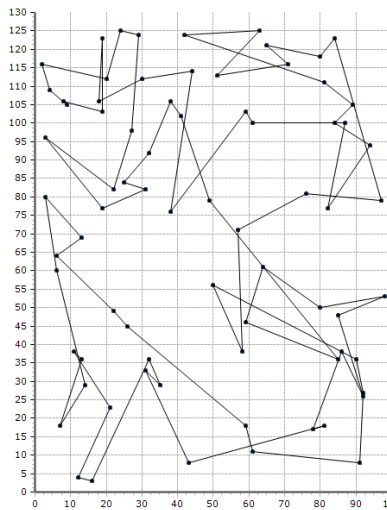


Abbildung 2-9: Zuf., Gen. 1.000

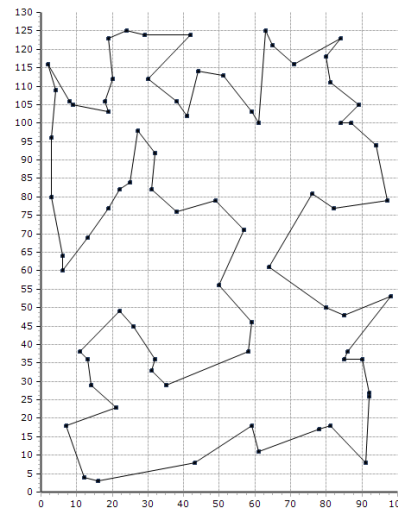


Abbildung 2-10: Zuf., Gen. 19.000

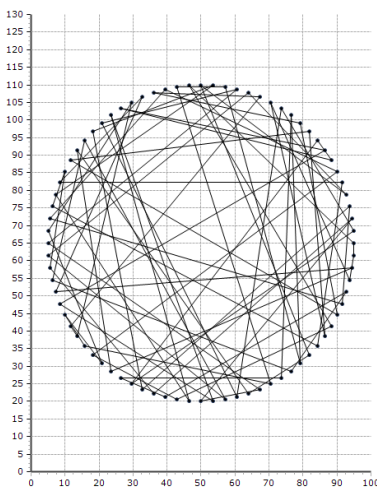


Abbildung 2-11: Kreis, Gen. 1

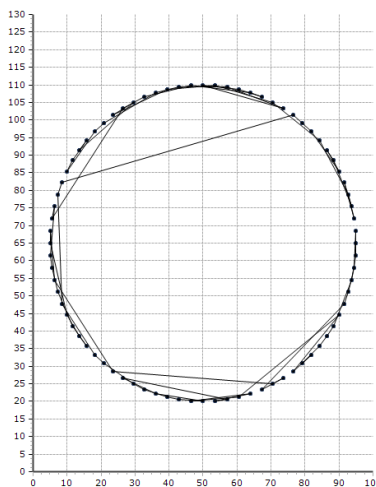


Abbildung 2-12: Kreis, Gen. 500

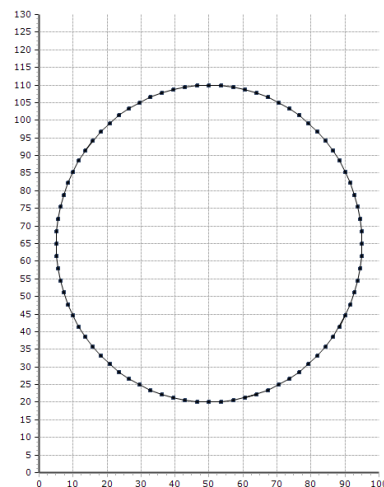


Abbildung 2-13: Kreis, Gen. 10.000

Die Kreisinstanz wurde zwecks Validierung definiert, da hier die optimale Route bekannt ist. Die Länge der optimalen Route beträgt

$$l = 2 \cdot n \cdot r \cdot \sin \frac{180^\circ}{n} \quad (2-11)$$

und entspricht dem Umfang eines n -Ecks. Die exakte Lösung für die zufällig verteilten Städte ist zunächst nicht bekannt. Sie kann hier nur „optisch“ überprüft werden. Beispielsweise darf der gewählte Weg sich nicht kreuzen und Zickzackrouten sollten vermieden

werden. Exakte Lösungen sind auch mit Methoden der ganzzahligen linearen Optimierung, beispielsweise Branch-and-Cut, möglich. Mit diesen lassen sich Instanzen beweisbar optimal lösen oder zumindest die Güte einer gefundenen Tour im Vergleich zu der exakten Lösung abschätzen.

Beide TSP-Testinstanzen konnten mit dem entwickelten Algorithmus bis zu einer Zahl von 70 - 80 Städten ($5,99 \cdot 10^{99} - 3,58 \cdot 10^{118}$ mögliche Kombinationen) in 10 - 30 Sekunden und 25.000 – 45.000 Evaluierungen gelöst werden. Bei Instanzen mit mehr als 80 Städten konnte die exakte Lösung in Minuten bis Stunden gelöst werden. Für eine Instanz mit einer Anzahl von 200 Städten und $1,97 \cdot 10^{372}$ Möglichkeiten findet der entwickelte Algorithmus innerhalb 5 Min – 10 Min eine Lösung mit einer Güte von 1,2 (die Strecke der gefundenen Lösung beträgt das 1.2 fache der Strecke der exakten Lösung). Für eine Instanz mit 300 und $5,1 \cdot 10^{611}$ Möglichkeiten findet er nach circa 20 Minuten eine Lösung mit der Güte von 2.

Den Grund für das schnelle Ansteigen der erforderlichen Rechenzeit bei mehr als 80 Instanzen soll ein Beispiel verdeutlichen, welches auch die unterschiedlichen Eigenschaften der in Abbildung 2-2 (Klassifizierung der Optimierungsmethoden) dargestellten Optimierungsmethoden verdeutlicht. Dort stellt Syrjakow (2005) das Durchsuchen des gesamten Lösungsraumes den heuristischen Methoden gegenüber. Die Berechnung aller möglichen Wege einer Instanz von 15 Städten würde auf dem in dieser Arbeit vorhandenen PentiumM 2 Mhz Prozessor ca. 44 Stunden dauern. Wird die Instanz um zwei Städte erweitert, also 16, beträgt die Rechenzeit 441 Tage, denn die Größe des Suchraums hängt überexponentiell von der Anzahl der Städte ab (siehe hierzu auch Abbildung 3-2).

2.7.4.1 Vergleich der Operatoren

Für den Vergleich der soeben beschriebenen Reproduktions- und Mutationsoperatoren wurde eine Testinstanz mit 100 ringförmig angeordneten Städten verwendet. Es existieren $4,67 \cdot 10^{157}$ mögliche Reisen. Die Optimierung wurde nach 500, 10.000 und 30.000 Generationen abgebrochen und die erreichten Qualitätswerte verglichen. Die exakte Lösung (der Umfang des 100-Ecks) beträgt 282,7 (Gleichung (2-11)). Für den Vergleich wurden die beiden Reproduktionsoperatoren in Kombination mit allen vier Mutationsoperatoren erprobt. Die Mittelwerte für die beiden Reproduktionsoperatoren sind in der Abbildung 2-14 auf der rechten Seite aufgetragen.

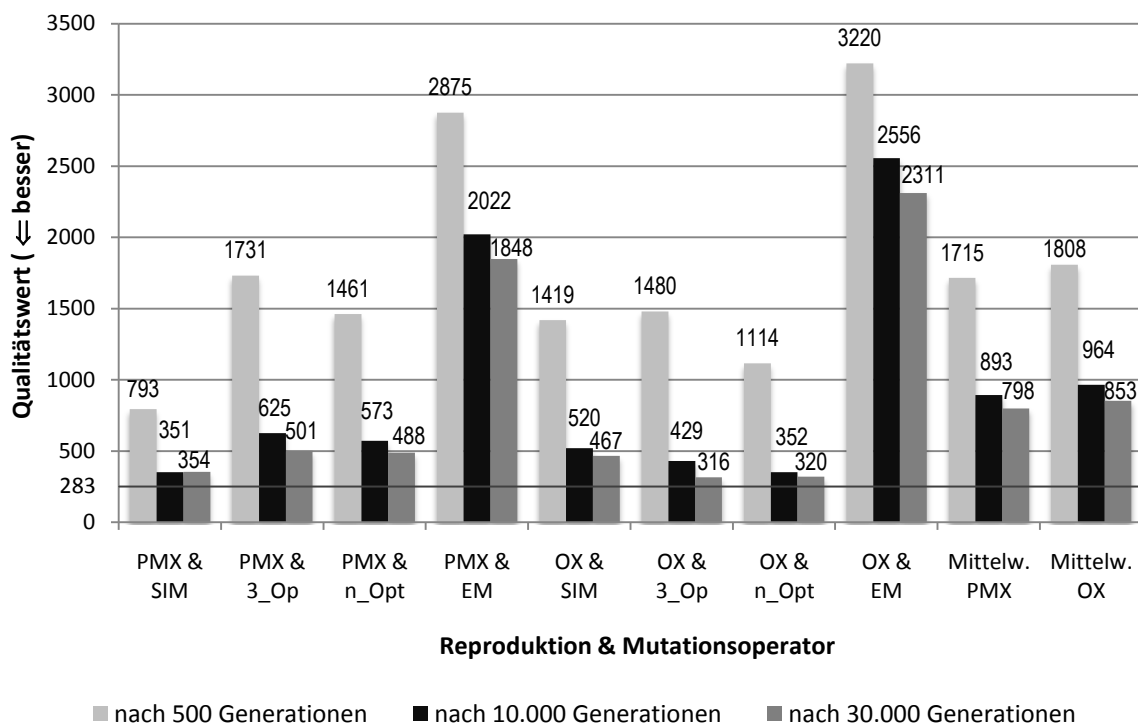


Abbildung 2-14: Vergleich der erreichten Qualitäten, exakte Lösung bei 283

Den besten Wert nach 30.000 Generationen erreichte die Kombination aus dem *OX* und dem *3Opt* Operator. Die gefundene Rundreise mit einer Länge von 316 entspricht dem 1,12 fachen der exakten Lösung. Ein unwesentlich schlechteres Ergebnis errichtet die Kombination aus *OX* und *nOpt* mit einer Rundreisenlänge von 320. Allerdings wurde von diesem Pärchen ein im Vergleich besseres Ergebnis (1114 zu 1480) nach 500 Generationen erreicht. Das beste Ergebnis nach 500 Generationen erreichte die Kombination aus *PMX* und *SIM*. Der *EM* Mutationsoperator lieferte in beiden Fällen die schlechtesten Ergebnisse.

Wie oben beschrieben, ist die Funktionsweise des *3Opt* und des *nOpt* Operators identisch. In der Praxis hat sich gezeigt, dass mit $n = 2$ durchaus Verbesserungen zu erzielen sind. Werte für $n > 3$ wurden erwiesen sich als kontraproduktiv.

2.7.5 Diskussion

Allen Ergebnissen ist gemein, dass der Optimierungsprozess bis zu einer Güte von 2 sehr schnell voranschreitet. Ist allerdings bereits eine gleichwertige Rundreise gefunden worden, wird es für den Algorithmus schwer die Schnittmarken an die richtige Stelle zu setzen. Wenn bei 100 Städten lediglich zwei Städte zu vertauschen sind, werden aufgrund

des stochastischen Prozesses viele Versuche benötigt, um an der richtigen Position des Strings zu schneiden und die Reihenfolge zu invertieren. Für eine zuverlässige Optimierung ist eine ausgewogene Kombination aus Mutations- und Reproduktionsprozessen erforderlich. Der Partially Mapped Crossover Reproduktionsoperator, welcher in Reihenfolge als auch Position der Städte eingreift, arbeitet gut mit dem Simple Inversion Mutationsoperator zusammen. Der Order Crossover Reproduktionsoperator legt gesteigerten Wert auf den Erhalt der Reihenfolge und weniger auf die Position der Städte. Er arbeitet besser mit den Translocation *3Opt* oder *nOpt* Mutationsoperatoren zusammen, welche im Gegensatz zum Simple Inversion Mutationsoperator nicht nur invertieren, sondern auch translokieren.

Abschließend kann gesagt werden, dass der entwickelte Algorithmus eine zuverlässige Grundlage bildet um ihn für die Hochwasserspeicheroptimierung einzusetzen.

3 Methodik der Hochwasserspeicheroptimierung

Ziel dieser Arbeit ist die multikriterielle Optimierung von Hochwasserspeichersystemen in einem Flusseinzugsgebiet. Die Frage nach der besten Lage für ein oder mehrere Hochwasserrückhaltebecken oder Polder in einem Einzugsgebiet ist eine kombinatorische Fragestellung. Zur Beantwortung dieser Frage, ist ein kombinatorischer Optimierungsalgorithmus erforderlich. Daher wird im Abschnitt 3.1 zunächst die in dieser Arbeit entwickelte nominell diskrete Hochwasserspeicheroptimierung vorgestellt.

Die zweite Fragestellung, die beantwortet werden soll, ist die Frage nach der Größe (Volumina) der Speicher und der optimalen konstanten Abgaben um den Scheitel eines Hochwasserereignisses maximal zu reduzieren. Zu diesem Zweck wird im Abschnitt 3.2 ein in dieser Arbeit entwickelter Memetischer Evolutionärer Algorithmus vorgestellt, der die Optimierung der Lage als auch der Volumina und der Abgaben der Speicher erlaubt.

3.1 Methodik der nominell diskreten Hochwasserspeicheroptimierung

Um diese Problemstellung zu lösen, muss der Genotyp in einer Form repräsentiert werden, welche die Anwendung von Reproduktions- und Mutationsprozessen erlaubt. In den folgenden Kapiteln werden die Art der Repräsentation und die entsprechenden Operatoren vorgestellt. Besonders berücksichtigt wird an dieser Stelle ein Vergleich der Lösungsräume zwischen den Problemdomänen *Hochwasserschutz* und dem *Problem des Handelsreisenden*.

3.1.1 Repräsentation der Hochwasserspeicher

Es vorausgesetzt, dass die Lage für geeignete Maßnahmen bekannt ist. Beispielsweise wurden an einem Gerinne sieben potenzielle Standorte identifiziert, (siehe Abbildung 3-1).

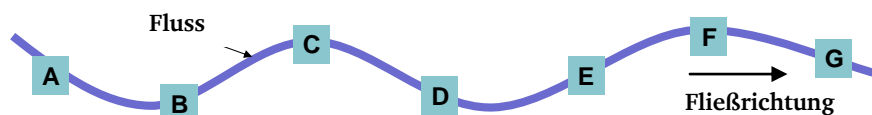


Abbildung 3-1: potenzielle Standorte für Hochwasserspeicher

Für den Fall, dass für jeden Standort drei Maßnahmen (M1- M3) und die Nullvariante (NV) zur Verfügung stehen, wird folgende Matrix (3-1) erstellt:

ID	A	B	C	D	E	F	G	
0	NV	NV	NV	NV	NV	NV	NV	
1	M1	M1	M1	M1	M1	M1	M1	(3-1)
2	M2	M2	M2	M2	M2	M2	M2	
3	M3	M3	M3	M3	M3	M3	M3	

Die Maßnahmen werden durch Integer Werte ersetzt. Das bedeutet, die Nullvariante wird mit 0 und die Maßnahmen mit den Zahlen 1-3 repräsentiert:

A	B	C	D	E	F	G	
0	0	0	0	0	0	0	
1	1	1	1	1	1	1	(3-2)
2	2	2	2	2	2	2	
3	3	3	3	3	3	3	

Eine Zahl von 0 – 3 definiert für jeden Standort die Maßnahme. Ein Vektor aus 8 Ziffern legt daher eindeutig alle Maßnahmen fest. Zu berücksichtigen bleibt, dass die ganzen Zahlen lediglich zur Identifikation der Maßnahme dienen, aber keine Rangfolge besitzen. Folgender ganzzahliger Vektor ist die **Repräsentation** des Hochwasserspeichersystems und definiert eine mögliche Kombination:

$$(1\ 3\ 0\ 2\ 1\ 2\ 1)$$
(3-3)

Weiter gilt es zu berücksichtigen, dass im Gegensatz zu der Permutation im symmetrischen Rundreiseproblem die Folge (3-3) eine andere Maßnahmenkombination ist als deren Inverse (3-4):

$$(1\ 2\ 1\ 2\ 0\ 3\ 1).$$
(3-4)

Daher wird hier genauer gesagt nicht mehr von Permutationen gesprochen, sondern von Variationen. Die Größe des Lösungsraumes entspricht der Anzahl N der Variationen und wird berechnet mit:

$$N = n^k$$
(3-5)

Mit: N : Anzahl der Variationen
 n : Anzahl der Maßnahmen pro Standort
 k : Anzahl der Standorte

Gleichung (3-5) ist gültig, wenn die Anzahl der Maßnahmen an jedem Standort n beträgt.

Dies ist in Bezug auf das Rundreiseproblem ein wesentlicher Unterschied. An einer Position im Genotyp des Hochwasserschutzproblems kann nicht wie beim Rundreiseproblem eine beliebige noch nicht bereiste Stadt gesetzt werden, sondern nur eine Maßnahme, die für diesen Standort gültig ist.

In der Praxis kann nicht davon ausgegangen werden, dass die Anzahl der Maßnahmen für jede Lage gleich groß ist. Daher ist die oben beschriebene Matrix ein Sonderfall. In der Praxis stehen für jede Lage zwei oder mehr Maßnahmen zur Verfügung, wenn die Nullvariante auch als Maßnahme betrachtet wird. Daher darf die Repräsentation des Hochwasserspeichersystems nicht aus einer Matrix (3-2) bestehen, sondern aus einem Vektorenfeld:

$$v \in D_i = \{\vec{d}_{i,1}, \dots, \vec{d}_{i,|D_i|}\}, i = 1, \dots, n_d \quad (3-6)$$

Das jedem Standort $x \in \Omega$ einen Vektor

$$v(x) \in D_j = \{d_{j,1}, \dots, d_{j,|D_j|}\}, j = 1, \dots, n_d \quad (3-7)$$

zuordnet.

Ein Beispiel sieht folgendermaßen aus:

A	B	C	D	E	F	G
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	2	2	2		2	2
3	3	3			3	3
4		4				4
		5				5

(3-8)

Dies muss bei der Wahl der Reproduktions- und Mutationsoperatoren berücksichtigt werden, da sie gültige Variationen erstellen müssen.

Ist die Anzahl der Maßnahmen an den Standorten unterschiedlich, gilt für die Anzahl der Möglichkeiten folgende Gleichung (3-9):

$$N = \prod_{i=0}^k n_i \quad (3-9)$$

In Abbildung 3-2 wird deutlich, dass sich der Lösungsraum (Anzahl der Möglichkeiten) im Vergleich zum Rundreiseproblem, bei gleicher Anzahl von Maßnahmen pro Lokation, nicht mehr überexponentiell sondern exponentiell entwickelt.

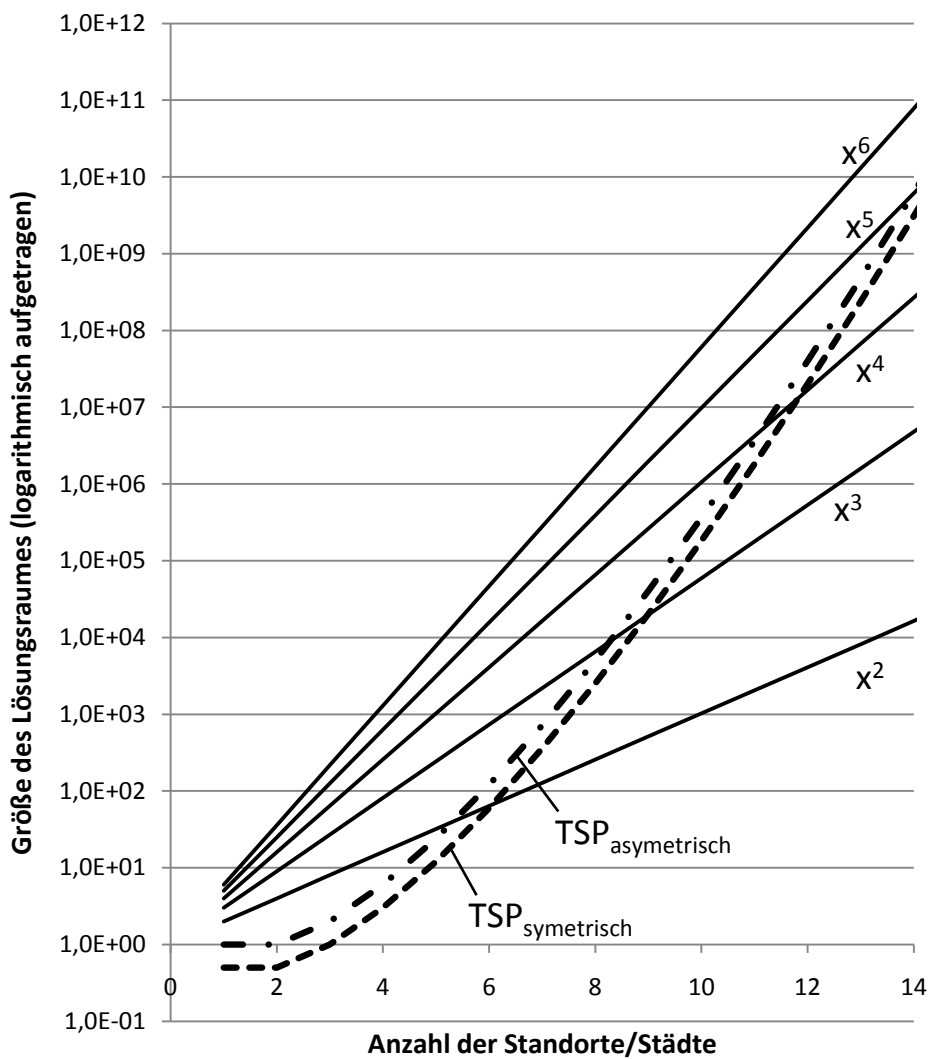


Abbildung 3-2: Logarithmischer Vergleich der Größe der Lösungsräume des Rundreiseproblems (TSP symmetrisch und asymmetrisch) und des hier vorliegenden Hochwasserschutzproblems in Abhängigkeit der Anzahl der Städte bzw. der Standorte

Die logarithmisch aufgetragene Ordinate verdeutlicht den überexponentiellen Verlauf. Auf der Abszisse ist die Anzahl der Städte beziehungsweise die Anzahl der Standorte aufgetragen. Die Exponenten der dargestellten Funktionen sind die Anzahlen der Maßnahmen an jedem Standort. Da es sich bei dem vorliegenden Problem in Abbildung 3-2 um Permutationen handelt, entwickelt sich der Lösungsraum bei einer kleinen Anzahl von Standorten schneller. Erst bei einer großen Zahl von Standorten / Städten wächst der Lösungsraum des Rundreiseproblems stärker. Geht man von der Annahme aus, dass es pro Standort 3 Alternativen und die Nullvariante gibt, so existieren für 7 Standorte $4^7 = 16.384$ Variationen. Der Lösungsraum des Rundreiseproblems ist bei 7 Städten $(7 - 1)!/2 = 360$ Permutationen groß. In diesem Fall ist die Zahl der Permutationen kleiner als die Zahl der Variationen. Bei 15 Standorten / Städten ist die Zahl der Variationen (1.073.741.824) kleiner als die Zahl der Permutationen und (43.589.145.600).

3.1.2 Reproduktions- und Mutationsoperatoren

Bei der Problemdomäne Hochwasserschutz handelt es sich um eine Variation mit Zurücklegen. Daher kann auf die klassischen genetischen Operatoren zugegriffen werden, denn der genetische Bitstring ist ebenfalls eine Variation mit Zurücklegen bei der n immer gleich 2 ist (0 oder 1 an jedem Allel) und k der Länge des Strings entspricht. Im Bereich der genetischen Algorithmen wird häufig der aus der Genetik stammende Begriff „Allel“ verwendet. Dieser bezeichnet eine mögliche Ausprägung eines Gens, das sich an einem bestimmten Ort (Lokalität) auf einem Chromosom befindet. Der Unterschied der hier gewählten Repräsentation zum genetischen Bitstring ist, das $n > 1$ und variabel sein darf.

Zur Optimierung des Hochwasserspeichersystems wurden folgende **nominell diskreten Reproduktionsoperatoren** implementiert:

- One Point Crossover (OPC)
- Two Point Crossover (TPC)
- k-Point Crossover (kPC)
- Uniform Crossover (UX)

Eine detaillierte Beschreibung der Reproduktionsoperatoren befindet sich in Anlage B - 1.

Auch für die Mutation werden Operatoren verwendet, die für bit-codierte Probleme entwickelt wurden. Es handelt sich in beiden Fällen um Operatoren aus dem Bereich der genetischen Algorithmen, die hier für $n > 1$ und variabel angepasst wurden. Bei den folgenden Mutations- und obigen Reproduktionsoperatoren wurde bei der Auswahl der Operatoren darauf geachtet, dass keine Reparaturmechanismen erforderlich sind. Beispielsweise könnten Operatoren eingesetzt werden, welche nicht berücksichtigen, dass n variabel ist. Diese würden im ersten Schritt einen Nachfahren erzeugen. Im zweiten Schritt muss überprüft werden, ob eine Maßnahme mit der Nr. 5 am ersten Allel gültig ist. Falls am ersten Allel nur 3 Maßnahmen zur Verfügung stehen, müsste eine „Reparatur“ des Genotyps erfolgen.

Folgende **nominell diskrete Mutationsoperatoren** wurden zur Lösung des Hochwasserspeicherproblems implementiert:

- Random Bit Value (RBV)
- Gene Insertion (GI)

Die Funktionsweise der nominell diskreten Mutationsoperatoren wird in der Anlage B - 2 detailliert beschrieben.

3.1.3 Prozedur des Algorithmus

Die Prozedur entspricht der Prozedur zur Rundreiseoptimierung (Algorithmus 3-1). Diese wurde um die Selektion mit der *Komma-Strategie* für die monokriterielle Optimierung und um die *multikriterielle Selektion* zur Lösung multikriterieller Probleme erweitert.

```

1 Initialisiere:    $P_0 = \{\vec{d}_1, \dots, \vec{d}_\mu\}$ 
2 Do While Not  Terminierungsbedingung
3   For i = 0 To  $\gamma - 1$ 
4     Reproduziere:  $P_i^* = \{\vec{d}_i^*\} = X P_i$ 
5     Mutiere:      $P_i^{**} = \{\vec{d}_i^{**}\} = \theta P_i^*$ 
6     For j = 0 To  $\lambda - 1$ 
7       Evaluiere:  $P_{ij}^{***} = \Phi\{\vec{d}_{ij}^{**}\}$ 
8     Next j
9     Selektiere:   $P_{i+1} = \mathcal{E}(P_i^{***} \cup P_i)$    (Plus-Strategie)
10    or
11    Selektiere:   $P_{i+1} = \mathcal{E}(P_i^{***})$          (Komma-Strategie)

```

12	or	
13	Selektiere:	$P_{i+1} = \mathcal{E}_M(P_i^{***})$ (Multikriterielle Selektion)
14	Next	i
15	Loop	

<i>mit:</i>	$d_i \in D_i = \{d_{i,1}, \dots, d_{i, D_i }\}$	<i>Nominaler Entscheidungsraum/Genotyp</i>
	$i = 1, \dots, n_d$	
	γ	<i>Anzahl der Generation.</i>
	μ	<i>Anzahl der Eltern.</i>
	λ	<i>Anzahl der Nachfahren.</i>
	X	<i>Reproduktionsoperator</i>
	θ	<i>Mutationsoperator</i>
	ϕ	<i>Evaluierungsfunktion (Simulationsmodell)</i>
	\mathcal{E}	<i>Selektionsoperator</i>
	\mathcal{E}_M	<i>Multikriterieller Selektionsoperator</i>
	P_i	<i>Population</i>
	$d_i = d_{i1}, \dots, d_{in}, \quad i \in \{1, 2, \dots, \mu\}$	<i>Vektor der nominalen Entscheidungsvariablen eines Elter.</i>
	P_i^* bis $P_i^{***}, \quad i \in \{1, 2, \dots, \gamma\}$	<i>Nachkommen einer Population der unterschiedlichen Stufen.</i>
	$\vec{d}_{ij}^* = d_{i1}^*, \dots, d_{in}^*$ bis	<i>Vektor der nominalen Entscheidungsvariablen eines Nachkommens</i>
	$\vec{d}_{ij}^{**} = d_{i1}^{**}, \dots, d_{in}^{**}$	<i>der unterschiedlichen Stufen.</i>
	$i \in \{1, 2, \dots, \gamma\}, j \in \{1, 2, \dots, \lambda\}$	

Algorithmus 3-1: Entwickelter evolutionsstrategischer kombinatorischer Algorithmus für nominell diskrete Zahlen

3.1.4 Erprobung

Zur Erprobung und Validierung wurden obige Operatoren in den nominell diskreten Algorithmus integriert. Der Testfall ist das später im Kapitel 6 beschriebene „Anwendungsbeispiel Erf“. Es wurden die Operatoren One Point Crossover (OPC), Two Point Crossover (TPC), k-Point Crossover (kPC) und Uniform Crossover UX getestet (Abbildung 3-3).

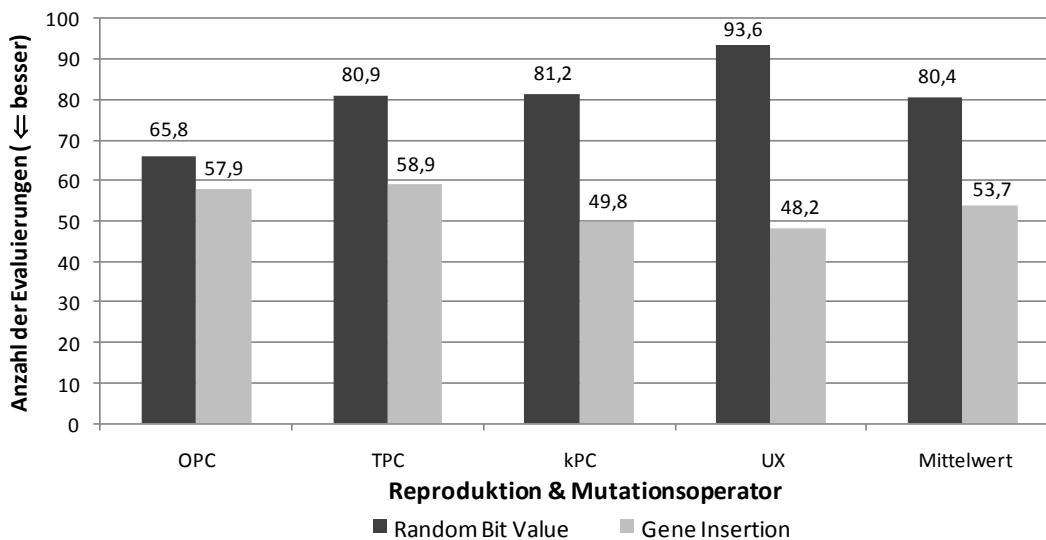


Abbildung 3-3: Vergleich der Operatoren für die Problemdomäne Hochwasserschutz

Es wurde erprobt, nach wie vielen Evaluierungen ein bestimmter Qualitätswert erreicht wurde. War das vorab ermittelte Optimierungsziel erreicht, wurde die Optimierung abgebrochen. Pro Kombination aus Reproduktions- und Mutationsoperator wurden 20 Optimierungsläufe durchgeführt und der Mittelwert gebildet.

3.1.5 Diskussion

Mit dem Zwischenschritt der Entwicklung des Algorithmus zur Optimierung des kombinatorischen Rundreiseproblems wurde das Ziel erreicht einen nominell diskreten Algorithmus zur Optimierung von Hochwasserspeichersystemen zu entwickeln. Die Prozedur und die softwareseitigen Rahmenbedingungen konnten anhand des Rundreiseproblems entwickelt und auf den Hochwasserschutz übertragen werden. Für die Problem- domäne Hochwasserschutz wurde eine geeignete Repräsentation entwickelt, welche die zu Optimierenden ganzzahligen nominalen Parameter enthält. Dabei handelt es sich um eine Variation mit Zurücklegen. Daher wurden entsprechende Operatoren implementiert, welche in der Lage sind, die Repräsentation zu Reproduzieren und zu Mutieren. Folgende Diskussion vergleicht die Operatoren und gibt Hinweise auf deren prinzipielle Funktions- weise und deren Eignung zur Lösung des Hochwasserspeichersystems.

Vergleich der Operatoren

Eindeutiges Ergebnis ist, dass die Gene Insertion Mutation besser abschneidet als die Random Bit Value Mutation (siehe Abbildung 3-3 „Mittelwert“). In jeder Kombination schnitt diese besser ab und im Mittel um ca 30 % besser als der Random Bit Value Operator. Am schlechtesten verhielt sich die Kombination aus Random Bit Value Mutation und Uniform Crossover Reproduktion, mit über 90 erforderlichen Evaluierungen. Die beste Kombination bildet die Gene Insertion Mutation und Uniform Crossover Reproduktion. Sie benötigt mit circa 50 die Hälfte der Evaluierungen.

Die Gene Insertion Mutation verändert lediglich einzelne Abschnitte des Nachkommens, während andere Bereiche unverändert bleiben. Dies ist wahrscheinlich der Grund, warum dieser Operator schneller arbeitet. Wenn die Mutationsrate nicht zu groß gewählt wird, bleiben viele qualitativ gute Gensequenzen erhalten, während bei der gleichen Mutationsrate und der Random Bit Value Mutation verteilt einzelne Allele auf dem Strang verändert werden. Dadurch werden viele Gensequenzen unterbrochen.

Die Ergebnisse der Reproduktionsoperatoren fallen weniger unterschiedlich aus. Insbesondere die One Point Crossover, Two Point Crossover und k-Point Crossover optimieren mit vergleichbarer Geschwindigkeit, da es sich um den gleichen Mechanismus handelt. Interessant ist, dass der Uniform Crossover Reproduktionsoperator mit der Gene Insertion Mutation die besten und mit der Random Bit Value Mutation die schlechtesten Ergebnisse liefert. Das zufällige Übertragen der Gene beider Eltern verändert den gesamten Strang. Wie bereits erwähnt verändert die Random Bit Mutation auch die gesamte Gensequenz. Dies führt dazu, dass ggf. gute neue Sequenzen, die aus den beiden Eltern geformt wurden wieder verändert werden. Dagegen können neu gebildete Bereiche bei der Gene Insertion unverändert bleiben.

Die Gene Insertion optimiert die Problemdomäne Hochwasserschutz vermutlich besser, da Kombinationen aus Maßnahmen in den Gensequenzen erhalten bleiben, die den Hochwasserscheitel sich für diesen Flussabschnitt gut reduzieren konnten. Dies ist nachvollziehbar, da Hochwasserschutzmaßnahmen nur einen begrenzten Wirkungsbereich haben.

Gwiazda (2006; 2007) stellt Hunderte von Reproduktions- und Mutationsoperatoren vor. Mit Sicherheit kann die Leistung des Algorithmus mit weiteren Operatoren noch erheblich gesteigert werden. Wie zum Beispiel Mutationsoperatoren mit Fehlerkorrektur oder selbst

adaptierende Operatoren. Allerdings wird darauf verzichtet, da das vorliegende Problem der Hochwasserspeicheroptimierung mit dem hier entwickelten Algorithmus bereits ausreichend schnell und sicher gelöst werden kann. Insgesamt kann festgehalten werden, dass alle Operatoren in der Lage sind die Hochwasserspeicher in Lage und Typ zu optimieren.

3.2 Methodik der Memetischen Hochwasserspeicheroptimierung

Der bisher entwickelte Algorithmus zur nominell diskreten Optimierung erlaubt es, die Lage und den Typ von Hochwasserspeichern zu optimieren. Allerdings sind Hochwasserspeicher Systemelemente mit modifizierbaren Parametern. Dies kann beispielsweise die Größe der Rückhaltebecken, die Steuerstrategie oder auch die Breite von Gerinnestrecken sein. Durch das Generieren weiterer Varianten mit abweichenden Parametereinstellungen ist es möglich, mit dem entwickelten Algorithmus einzelne Parametereinstellungen zu vergleichen. Mehr als das Vergleichen einzelner Parametereinstellungen ist auf diese Weise nicht möglich.

Gerade bei der Kombination von Speichern hat eine etwaige Steuerung oder die Quantifizierung der Größe (Volumina) der Maßnahme erheblichen Anteil an der Zielerreichung, da die Maßnahmen interagieren. Daher ist es Ziel des im Folgenden beschriebenen memetischen Algorithmus diese reellwertigen Größen bei der Optimierung zu berücksichtigen. Zu diesem Zweck sind entsprechende reellwertige Operatoren erforderlich.

Dabei muss berücksichtigt werden, dass die enthaltenen nominal diskreten Optimierungsparameter einen inhomogenen Lösungsraum erzeugen. Die Gradientenwege verlaufen nicht kontinuierlich (Gantovnik u. a., 2003). Die nominell diskrete Optimierung kann als der reellwertigen übergeordnet betrachtet werden. Des Weiteren sind die reellwertigen Parameter von den nominell diskreten Parametern abhängig. Jeder durch die nominal diskreten Parameter definierte Unterlösungsraum besitzt daher ein eigenes globales Optimum. Diese Optima werden durch die reellwertigen Anteile definiert. Daher enthält der Gesamtlösungsraum mindestens $\prod_{i=0}^k n_i$ lokale Minima.

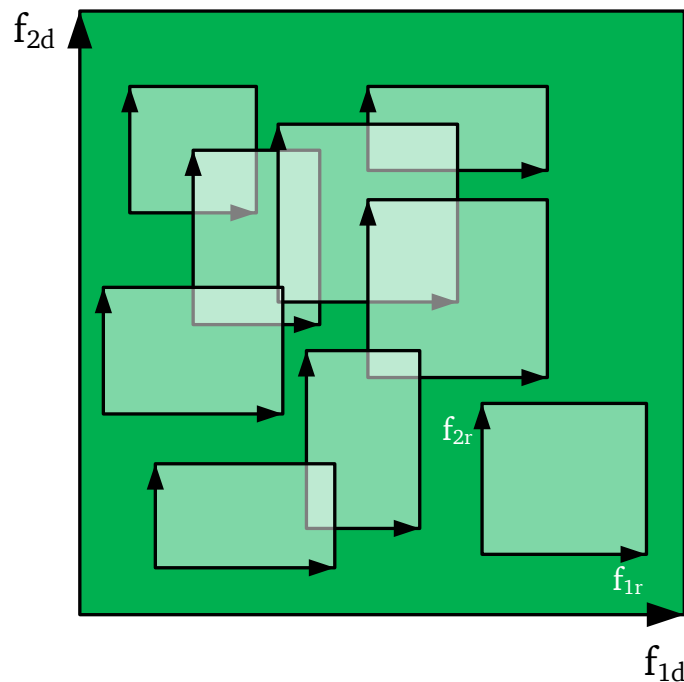


Abbildung 3-4: Überlagerung der Lösungsräume

Abbildung 3-4 zeigt die Überlagerung der Lösungsräume mit zwei Dimensionen. Die Achsen $f_{1diskret}$ und $f_{2diskret}$ spannen den übergeordneten nominellen Lösungsraum auf. Dieser beinhaltet $\prod_{i=0}^k n_i$ von f_{1reell} und f_{2reell} aufgespannte Sublösungsräume. Die diskret globale Suche wird lokal von einer reell globalen Suche unterstützt.

Die Architektur des Algorithmus muss daher so gestaltet sein, dass er mit diesem inhomogenen Lösungsraum umgehen kann. Bestenfalls ist er in der Lage die reellwertigen Teile soweit anzupassen, dass beim kombinatorischen Anpassen im Wesentlichen die bereits gefunden lokalen Minima verglichen werden.

Sogenannte hybride Algorithmen können aus zwei oder mehr evolutionären Algorithmen bestehen oder evolutionäre Algorithmen werden mit anderen Optimierungsverfahren wie beispielsweise Gradientenverfahren, Simulated Annealing, Hill Climbing oder Dynamischer Programmierung kombiniert. Ziel dabei ist es, die Schwächen eines einzelnen Algorithmus auszugleichen und sie so besser an die Problemdomäne anzupassen oder, wie in diesem Fall, den Anwendungsbereich zu erweitern.

Für diese funktionale Erweiterung ist es nicht erforderlich neue Reproduktions- und Mutationsoperatoren einzusetzen. Die Repräsentation besteht aus einem nominell diskreten Teil und einem reellwertigen Teil. Für die nominell diskreten Teile werden die in

dieser Arbeit entwickelten Operatoren eingesetzt. Für die reellen Anteile werden aus den Arbeiten Muschalla (2006) und Huebner (2003) verwendet. Damit beide Typen von Operatoren zusammenarbeiten ist eine neue Prozedur und ein neuer Selektionsoperator erforderlich, um die reellwertigen und nominell diskreten Anteile der Repräsentation geeignet zu selektieren.

Der hier entwickelte Algorithmus wurde als **Memetischer Evolutionärer Algorithmus** klassifiziert. Der Begriff des »Meme«, geht auf den griechischen Begriff „mimeme“ – Imitation, zurück. Er bezeichnet Verhaltenselemente, welche sich im Gegensatz zu Genen individuell ändern können, indem sie beispielsweise durch Nachahmung erworben werden. Dies Nachahmung erfolgt im Rahmen der Kultur, in der das Individuum sich befindet, und ist der biologischen Evolution nachgeordnet. Die nominell diskrete Optimierung kann als übergeordnete globale Suche betrachtet werden und die Optimierung der reellen Parameter als lokale Suche in einem Sublösungsraum. Dies ist vergleichbar mit einer Kombination aus biologischer Evolution und anschließender kultureller Evolution. Näheres zur kulturellen Evolution ist in der Anlage C „Klassifizierung des memetischen Algorithmus“ zu finden.

3.2.1 Voraussetzung zur memetischen Optimierung

Die Ausgangssituation zur Entwicklung des Memetischen Evolutionären Algorithmus stellt sich folgendermaßen dar: Es liegen ein kombinatorischer evolutionstrategischer Algorithmus zur nominell diskreten Optimierung und ein evolutionstrategischer Algorithmus zur reellwertigen Parameteroptimierung vor. Die Programmiersprache ist einheitlich VB.NET, wobei der reellwertige Algorithmus ursprünglich in Visual Basic 6 implementiert und später migriert wurde. Beide greifen auf dieselben Funktionen zur Steuerung des Simulationsmodells und zum Auswerten der Ergebnisse zu, sowie teilweise auf die gleichen Steuerungsdateien für die Algorithmen (z. B. die Zielfunktionen). Die Ablaufkontrolle zur Evaluierung der Nachfahren, dem Ermitteln der Eltern und die Schleifen für Generationen und Populationen waren bisher getrennt. Ebenfalls verschieden sind zwingendermaßen die Operatoren zur Reproduktion und Mutation. Bei der Selektion wird teilweise auf bereits implementierte Funktionen des reellwertigen Algorithmus zurückgegriffen. Für monokriterielle Problemstellungen müssen die Individuen einer Generation „lediglich“ ihres Zielfunktionswertes nach sortiert werden. Im Falle einer multikriteriellen Problemstellung ist

dies, wie in Kapitel 2.4.4 „Mono- und multikriterielle Optimierung“ erläutert, nicht mehr so einfach möglich, da es kein einzelnes Optimum gibt, sondern eine optimale Lösungsmenge, die Paretofront. Ein Sortiermechanismus, der die nicht dominierten Lösungen ermittelt, wurde von (D. Muschalla, 2006) auf Basis der Arbeiten von Srinivas und K. Deb (1995) und K. Deb et al. (2000) implementiert und erfolgreich angewandt. Um die Funktionen nicht erneut entwickeln zu müssen, wurden diese generalisiert, sodass alle Algorithmen das sogenannte Non-Dominated-Sorting ausführen können.

Die folgenden Unterkapitel gehen insbesondere auf die Punkte *Repräsentation des Hochwasserspeichersystems*, *Selektion* und auf die hier eigens entwickelte *Prozedur des memetischen Algorithmus* ein. Die neue Prozedur charakterisiert den memetischen Algorithmus und ist erforderlich, um die vorhandenen Operatoren für die nominell diskreten und reellwertigen Anteile der Repräsentation anzuwenden. Daher werden drei Varianten für eine geeignete Prozedur vorgestellt und diskutiert.

3.2.2 Reproduktions- und Mutationsoperatoren

Für den memetischen Algorithmus ist es nicht erforderlich weitere Reproduktions- und Mutationsmechanismen einzuführen. Für die nominell diskrete Repräsentation werden die Operatoren aus Kapitel 3.1.2 „Reproduktions- und Mutationsoperatoren“ eingesetzt. Für die reellwertige Repräsentation werden weiterhin die Operatoren aus Kapitel 2.6.2 „Reproduktions- und Mutationsoperatoren“ verwendet.

3.2.3 Selektionsoperator

Für den Selektionsprozess wurde bislang im Falle von multikriteriellen Problemdomänen der vorhandene Selektionsoperator \mathcal{E}_M verwendet. Dieser wurde modifiziert, sodass er mit reellwertigen, als auch mit nominell diskreten Individuen umgehen kann. Bei der monokriteriellen Optimierung ist die Selektion \mathcal{E} durch das Sortieren anhand des Qualitätswertes möglich, wobei - wie oben bereits erwähnt - zwischen der Plus-Strategie und der Komma-Strategie unterschieden werden muss.

Im folgenden Kapitel werden die im Rahmen dieser Arbeit entwickelten Entwürfe für den memetischen Algorithmus vorgestellt. Dabei musste bei der letztendlich gewählten Architektur zusätzlich ein neuer mehrstufiger Selektionsoperator \mathcal{E}_{mem} eingeführt werden.

3.2.4 Prozeduren und Repräsentationen der Algorithmen

Für den memetischen Algorithmus muss der nominal diskrete Algorithmus 3-1 mit dem parametrischen Algorithmus 2-1 für die kontinuierlichen Parameter verknüpft werden. Dabei muss berücksichtigt werden, dass Reproduktions-, Mutations-, und Selektionsoperatoren richtig kombiniert werden, um sowohl die nominellen, als auch die reellen Parameter zu entwickeln. Daher werden hier zunächst einige Grundüberlegungen hinsichtlich der Architektur memetischen Algorithmus vorgestellt. Zur Diskussion steht:

1. die sequenzielle Optimierung,
2. die parallele Optimierung mit allen reellen Parametern und
3. die parallele Optimierung mit selektierten reellen Parametern.

Des Weiteren wird für jede dieser Prozeduren die zugehörige erforderliche Repräsentation vorgestellt.

3.2.4.1 Sequenzielle Optimierung

In diesem Fall würde zunächst die Kombination der Maßnahmen optimiert und in der Folge die Parameter der Maßnahmen. Die Optimierung der Kombination erfolgt mit den händisch oder zufällig gewählten Startparametern der einzelnen Maßnahmen. Für die beste Kombination werden im Anschluss die Parameter optimiert. Im Wesentlichen handelt es sich in diesem Fall um zwei voneinander abhängige Optimierungsläufe.

```
1 Initialisiere:  $P_0 = \{\vec{d}_0(k)_1, \dots, \vec{d}_0(k)_\mu\}$ 
2 Do While Not Terminierungsbedingung
3   For i = 0 To  $\gamma - 1$ 
4     Reproduziere nominal:  $P_{i(N)}^* = \{\vec{d}(k)_i^*\} = X_N P_{i(N)}$ 
5     Mutiere nominal:  $P_{i(N)}^{**} = \{(\vec{d}(k))_i^{**}\} = \Theta_N P_{i(N)}^*$ 
6     For j = 0 To  $\lambda - 1$ 
7       Evaluiere:  $P_{ij}^{***} = \Phi\{\vec{d}(k)_{ij}^{**}\}$ 
8     Next j
9     Selektiere:  $P_{i+1} = \mathcal{E}(P_i^{***} \cup P_i)$  (Plus-Strategie)
10    or
11    Selektiere:  $P_{i+1} = \mathcal{E}(P_i^{***})$  (Komma-Strategie)
12    or
13    Selektiere:  $P_{i+1} = \mathcal{E}_M(P_i^{***})$  (Multikriterielle Selektion)
```


14 **Next i**

15 **Loop**

16 Initialisiere: $P_0 = \left\{ \begin{array}{l} \vec{r}_1^0(m), \dots, \vec{r}_\mu^0(m) \\ \vec{\delta}_1, \dots, \vec{\delta}_\mu \end{array} \right\}$

17 **Do While Not** Terminierungsbedingung

18 **For i = 0 To** $\gamma - 1$

19 **For j = 0 To** $\lambda - 1$

20 Reproduziere reell: $P_{ij(R)}^* = \left\{ \begin{array}{l} \vec{r}_{ij}^*(m) \\ \vec{\delta}_{ij}^* \end{array} \right\} = X_R P_{ij(R)}$

21 Mutiere reell: $P_{ij(R)}^{**} = \left\{ \begin{array}{l} \vec{r}_{ij}^{**}(m) \\ \vec{\delta}_{ij}^{**} \end{array} \right\} = \Theta_R P_{ij(R)}^*$

22 Evaluieren: $P_{ij}^{***} = \Phi(\vec{r}_{ij}^{**}(m))$

23 **Next j**

24 Selektiere: $P_{i+1} = \Xi(P_i^{***} \cup P_i)$ (Plus-Strategie)

25 or

26 Selektiere: $P_{i+1} = \Xi(P_i^{***})$ (Komma-Strategie)

27 or

28 Selektiere: $P_{i+1} = \Xi_M(P_i^{***})$ (Multikriterielle Selektion)

29 **Next i**

30 **Loop**

mit: $d_i \in D_i = \{d_{i,1}, \dots, d_{i, D_i }\}, i = 1, \dots, n_d$	<i>Nominaler Entscheidungsraum/Genotyp</i>
$r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r$	<i>Reeller Entscheidungsraum/Genotyp</i>
$\vec{d}(k)$	<i>Vektor der nominalen Entscheidungsvariablen eines Elter in Abhängigkeit der reellen Variablen</i>
$\vec{r}(m)$	<i>Vektor der reellen Entscheidungsvariablen in Abhängigkeit der nominalen Variablen.</i>
X_N	<i>Reproduktionsoperator für nominale Variablen</i>
X_R	<i>Reproduktionsoperator für reelle Variablen</i>
Θ_N	<i>Mutationsoperator für nominale Variablen</i>
Θ_R	<i>Mutationsoperator für reelle Variablen</i>
Ξ	<i>Selektionsoperator</i>
Ξ_M	<i>Multikriterieller Selektionsoperator</i>
Φ	<i>Evaluierungs-/Simulationsoperator</i>
P_i	<i>Population</i>
$P_{i(N)}$	<i>Nominaler Teil der Population</i>

Algorithmus 3-2: Sequenzielle Prozedur des MEA

Repräsentation der sequenziellen Optimierung

Die Repräsentation für die sequenzielle Optimierung würde im ersten Schritt der Optimierung aus dem ganzzahligen Vektorenfeld \mathbf{v} (3-6) aus Kapitel 3.1.1 bestehen:

$$\mathbf{v} \in D_i = \{d_{i,1}, \dots, d_{i,|D_i|}\}, i = 1, \dots, n_d \quad (3-10)$$

Ein Beispiel sieht folgendermaßen aus:

$$(1 \ 3 \ 0 \ 2 \ 1 \ 2 \ 1) \quad (3-11)$$

welcher ebenso wie in der rein nominell diskrete Optimierung eine gültige Variation darstellen muss.

Im Zweiten Schritt würden die zu dieser Maßnahmenkombination zugehörigen reellen Parameter ermittelt und in einem reellwertigen Teilvektor \vec{r}_t zusammengefasst:

$$\vec{r}_t \subseteq \vec{r} \quad (3-12)$$

Ein Beispiel sieht folgendermaßen aus:

$$(1,3 \ 2,5 \ 83 \ 17 \ -154 \ 0,001) \quad (3-13)$$

Dieser würde nur die reellen Parameter der bereits ermittelten Repräsentation enthalten.

3.2.4.2 Parallele Optimierung mit allen reellen Parametern

Das Evolutionieren der Kombination und der Parameter erfolgt hier parallel. Die Prozesse der Reproduktion und Mutation wurden in jeder Generation auf beide Repräsentationen angewandt. Die Selektion erfolgt einmal pro Generation.

-
- 1 Initialisiere: $P_0 = [\vec{d}, \vec{r}]_1, \dots, [\vec{d}, \vec{r}]_\mu$
 - 2 **Do While Not** Terminierungsbedingung
 - 3 **For** $i = 0$ **To** $\gamma - 1$

```

4      Reproduziere nominal:  $P_{i(N)}^* = \vec{d}_i^* = X_N P_{i(N)}$ 
5      Reproduziere reell:    $P_{i(R)}^* = \vec{r}_i^* = X_R P_{i(R)}$ 
6      Mutiere nominal:      $P_{i(N)}^{**} = \vec{d}_i^{**} = \Theta_N P_{i(N)}^*$ 
7      Mutiere reell:        $P_{i(R)}^{**} = \vec{r}_i^{**} = \Theta_R P_{i(R)}^*$ 
8      For j = 0 To  $\lambda - 1$ 
9          Evaluiere:        $P_{ij}^{***} = \Phi[\vec{d}_i, \vec{r}_i]_{ij}^{**}$ 
10     Next j
11     Selektiere:           $P_{i+1} = \Xi(P_i^{***} \cup P_i)$       (Plus-Strategie)
12     or
13     Selektiere:           $P_{i+1} = \Xi(P_i^{***})$           (Komma-Strategie)
14     or
15     Selektiere:           $P_{i+1} = \Xi_M(P_i^{***})$         (Multikriterielle Selektion)
16     Next i
17 Loop

```

<i>mit:</i> $d_i \in D_i = \{d_{i,1}, \dots, d_{i, D_i }\}, i = 1, \dots, n_d$	<i>Nominaler Entscheidungsraum/Genotyp</i>
$r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r$	<i>Reeller Entscheidungsraum/Genotyp</i>
$X_N(\mu)$	<i>Reproduktionsoperator für nominale Variablen</i>
$X_R(\mu)$	<i>Reproduktionsop. für reelle Variablen</i>
Θ_N	<i>Mutationsoperator für nominale Variablen</i>
Θ_R	<i>Mutationsoperator für reelle Variablen</i>
Ξ	<i>Selektionsoperator</i>
Ξ_M	<i>Multikriterieller Selektionsoperator</i>
Φ	<i>Evaluierungs-/Simulationsoperator</i>
P_i	<i>Population</i>
$P_{i(N)}$	<i>Nominaler Teil der Population</i>
$P_{i(R)}$	<i>Reeller Teil der Population</i>

Algorithmus 3-3: Parallele Prozedur des MEA mit allen reellen Parametern

Repräsentation der parallelen Optimierung mit allen reellen Parametern

Die Repräsentation für die parallele Optimierung besteht aus dem Vektorfeld \mathbf{v} das die ganzzahligen Parameter und dem Vektor \vec{r} der reellen Parameter enthält:

$$\mathbf{v} \cup \vec{r} \tag{3-14}$$

Ein Beispiel sieht folgendermaßen aus:

$$(1\ 3\ 0\ 2\ 1\ 2\ 1\ | 1,3\ 2,5\ 83\ 17\ -154\ 0,1\ 3\ -8\ 24\ 12\ 1975) \quad (3-15)$$

Dabei wäre der reellwertige Anteil im Vergleich zu obigen reellwertigen Teilvektor (3-12) wesentlich größer, da alle reellen Anteile darin enthalten sein müssten.

3.2.4.3 Parallele Optimierung mit selektierten reellen Parametern

Das Evolutionieren der Kombination und der reellen Parameter erfolgt hier parallel. Allerdings werden die realparametrischen Reproduktions- und Mutationsprozesse nur auf die reellen Parameter angewendet, die in der gewählten Kombination des Individuums erforderlich sind. Die Selektion erfolgt einmal für die für die nominell diskrete Anteil und einmal für reellen Anteile der Repräsentation.

```

1  Initialisiere:       $P_0 = [\vec{d}, \vec{r}]_1, \dots, [\vec{d}, \vec{r}]_\mu$ 
                        $P_{mem} = \{\}$ 
2  Do While Not    Terminierungsbedingung
3      For i = 0 To  $\gamma - 1$ 
4          Reproduziere nominal:  $P_i^* = \vec{d}_i^* = X_N P_i$ 
5          Mutiere nominal:       $P_i^{**} = \vec{d}_i^{**} = \Theta_N P_i^*$ 
6          For n = 0 To  $s - 1$ 
7              Selektiere:       $P_{mem,n} = \Xi_{mem} \cup_{p=0}^{p=\tau} P_{i-p}$ 
8              or
9              Selektiere:       $P_{mem,n} = \Xi_{mem,M} \cup_{p=0}^{p=\tau} P_{i-p}$ 
10             Reproduziere reell:  $P_{in}^* = \vec{r}_{i,n}^* = X_R(\mu) P_{mem,n}$ 
11             Mutiere reell:      $P_{in}^{**} = \vec{r}_{i,n}^{**} = \Theta_R P_{in}^*$ 
12         Loop
13         For j = 0 To  $\lambda - 1$ 
14             Evaluiere:       $P_{ij}^{***} = \Phi([\vec{d}, \vec{r}]_{ij}^{**})$ 
15         Next j
16         Selektiere:       $P_{i+1} = \Xi(P_{ij}^{***} \cup P_i)$     (Plus-Strategie)
17         or
18         Selektiere:       $P_{i+1} = \Xi(P_{ij}^{***})$           (Komma-Strategie)
19         or
20         Selektiere:       $P_{i+1} = \Xi_M(P_{ij}^{***})$         (Multikriterielle Selektion)
21     Next i

```

22 Loop

<i>mit:</i>	$d_i \in D_i = \{d_{i,1}, \dots, d_{i, D_i }\}, i = 1, \dots, n_d$	<i>Nominaler Entscheidungsraum/Genotyp</i>
	$r_i \in [r_i^{\min}, r_i^{\max}] \subset \mathbb{R}, i = 1, \dots, n_r$	<i>Reeller Entscheidungsraum/Genotyp</i>
	s	<i>Größe des nominalen Vektors / Anzahl der Maßnahmen</i>
	P_{mem}	<i>Memory Population</i>
	X_N	<i>Reproduktionsop. für nominale Variablen</i>
	X_R	<i>Reproduktionsop. für reelle Variablen</i>
	θ_N	<i>Mutationsoperator für nominale Variablen</i>
	θ_R	<i>Mutationsoperator für reelle Variablen</i>
	\mathcal{E}	<i>Selektionsoperator</i>
	\mathcal{E}_M	<i>Multikriterieller Selektionsoperator</i>
	\mathcal{E}_{mem}	<i>Memory Selektionsoperator</i>
	Φ	<i>Evaluierungs-/Simulationsoperator</i>
	P_i	<i>Population</i>

Algorithmus 3-4: Parallele Prozedur mit selektierten reellen Parametern

Repräsentation der parallelen Optimierung mit selektierten reellen Parametern

Sollen die reellen Teile selektierbar sein, muss innerhalb der Repräsentation eine Information vorliegen, die definiert, welche reellen Parameter zu einer Maßnahme gehören. Daher muss jedem Standort (A, B oder C) und jeder Maßnahme ein Vektor aus reellen Parametern angehängt werden. Die Repräsentation für die parallele Optimierung mit selektierten Parametern besteht also aus dem Vektorfeld \mathbf{v} und dem reellen vektor \vec{r}_t :

$$\mathbf{v} \cup \vec{r}_t \quad (3-16)$$

Für den gilt:

$$\vec{r}_t \subseteq \vec{r} \quad (3-17)$$

Ein Beispiel sieht folgendermaßen aus:

A	B	C	
0 (1,5 20)	0 (2,3)	0 (0,8 3 5,5)	(3-18)
1 (3 5,6 112)	1 (56)	1 (3 5)	
2 (8 3)	2 (0,1 -3 5)	2	
3 (9 3)		3 (75 12 4)	

In (3-18) sind im jedem Standortvektor (A, B oder C) auf der linken Seite in Fett die möglichen Maßnahmen dargestellt. Rechts neben jeder Maßnahme steht in Klammern der reellwertige Vektor. Dabei kann die Länge des Vektors verschieden sein und jede Maßnahme eine andere Anzahl reeller Parameter beinhalten. Mit dieser Information können nach dem Mutieren der nominellen diskreten Nachfahren die zugehörigen reellwertigen Anteile ermittelt und evolutioniert werden.

Der \mathcal{E}_{mem} Selektionsoperator für die parallele Optimierung mit selektierten reellen Parametern

Zu diesem Zweck wurde der \mathcal{E}_{mem} Operator für die memetische Optimierung der Hochwasserspeicher entwickelt. In obigen Algorithmus 3-4 wird eine zweistufige Selektion vorgenommen, wobei die Zweite in Abhängigkeit der Ersten erfolgt. In der ersten Stufe werden mit dem Selektionsoperator \mathcal{E} beziehungsweise im multikriteriellen Fall \mathcal{E}_M die Maßnahmen nominell selektiert. Damit ist die Maßnahme an jeder Position definiert aber die neuen reellen Parameter noch unbekannt.

In der zweiten Stufe wird pro Allel eine monokriterielle \mathcal{E}_{mem} oder $\mathcal{E}_{mem,M}$ multikriterielle Selektion durchgeführt. Da an jeder Position mehrere Hochwasserspeicher platziert werden können liegen in einer Generation nicht für alle Maßnahmen reelle Parameter vor. Daher wurde der Speicher P_{mem} eingeführt. Dieser bewahrt die Eltern aus mehreren Generationen. Der Selektionsoperator \mathcal{E}_{mem} selektiert dann aus den letzten τ Populationen $P_{mem} = \bigcup_{p=0}^{p=\tau} P_{i-p}$ für jede aktive Maßnahme die Evaluierungsergebnisse, welche die größte Ähnlichkeit für einen bestimmten Bereich der nominalen Kombination aufweisen.

Berücksichtigung der physikalischen Eigenschaften bei der Selektion

Der Operator \mathcal{E}_{mem} selektiert standardmäßig Individuen aus dem Speicher P_{mem} die die gleiche Maßnahme an einem bestimmten Allel aktiviert haben. Zusätzlich wurde eine Methode entwickelt, die die physikalische Eigenschaften des Hochwasserablaufs gewinnbringend integriert. Wie oben bereits erwähnt besitzen Hochwasserschutzmaßnahmen nur

begrenzten Einfluss auf den Hochwasserablauf. Im Einzugsgebiet der Erft haben beispielsweise die meisten Maßnahmen einen lokalen bis regionalen Einflussbereich. Des Weiteren sind die Wirkungszusammenhänge bekannt. Das erforderliche Retentionsvolumen einer Maßnahme hängt davon ab, wie viel Volumen von dem oder den oberhalb liegenden Maßnahmen bereits aufgenommen werden konnte. Umgekehrt kann die Frage gestellt werden, wie viel Volumen noch erforderlich ist, wenn ein unterhalb liegendes Becken ein bestimmtes Retentionsvolumen bereithält. Zwar ist es nicht möglich, physikalische Gleichungen in einen Evolutionären Algorithmus einzubinden, denn das würde die Prinzipien verletzen, da aber ein physikalischer Zusammenhang besteht, kann der Algorithmus diesen bei der Selektion berücksichtigen. Bei der \mathcal{E}_{mem} Selektion kann daher eingestellt werden welche Individuen bevorzugt werden, indem verglichen wird, welche Maßnahmen oberhalb oder unterhalb des betrachteten Allels aktiviert wurden.

Mit der Einstellung **Two-Locations-Up** werden Individuen bevorzugt die am aktuellen Allel und an den beiden vorausgehenden Positionen die gleiche Maßnahmen aktiviert hatten. Bei der **One-Location-Up** wird die aktuelle und die vorausgehende Position betrachtet. Die Einstellung **This-Location** ist die Standardeinstellung und betrachtet nur ein Allel. Bei der Einstellung **One-Location-Down** wird das aktuelle und das folgende Allel betrachtet und bei **Two-Locations-Down** entsprechend die beiden folgenden Allele.

Ziel dieser zusätzliche Einstellung für die Selektion ist es einen maßgeschneiderten Pool von Individuen für die Reproduktion zu erhalten.

3.2.5 Vergleich und Diskussion

Für die Optimierung von Hochwasserspeichern wird die gewählte Maßnahme durch ganzzahlige nominal diskrete Parameter repräsentiert und Designparameter (Speichervolumen und konstante Abgaben) durch reelle Parameter. Zu berücksichtigen ist, dass die Zahl der zu variierenden reellen Parameter von den nominal diskreten Parametern abhängig ist. Es besteht eine starke Kausalität zwischen den reellen und diskreten Parametern. Daher haben nicht immer alle reellen Parameter in jedem Evaluierungsschritt Einfluss auf das Ergebnis der Optimierung, sondern nur diejenigen, die bei der aktuellen Auswahl nominal diskreter Parameter aktiviert werden. Dies führt zu dem – wie bereits in der Einführung dieses Kapitels beschriebenen - sehr inhomogenen Lösungsraum. Und zum anderen zu einer sequenziellen Optimierung, bei der erst die nominell diskreten Parameter optimiert

werden sollten. Entweder muss zuerst die rein nominelle Lösung ermittelt werden und dann die zugehörige reellwertige Lösung oder es muss innerhalb jeder Generation erst der nominelle Anteil evolutioniert werden und in der Folge die abhängigen reellwertigen Anteile. Die drei zur Auswahl stehenden Architekturen gehen dabei unterschiedlich vor:

Zu 1. Sequenzielle Optimierung: Bei der sequenziellen Optimierung wird zunächst durch Variation der Kombination mit dem nominalen Reproduktionsoperator X_N und dem Mutationsoperator Θ_N der nominal diskrete Lösungsraum untersucht. Dabei werden die reellen Startparameter $P_0 = \{r_0(m)_1, \dots, r_0(m)_\mu\}$ nicht verändert. Dies führt dazu, dass bei diesem Verfahren zwar der Gesamtlösungsraum Λ untersucht wird, in den Sublösungsräumen $\Lambda_{S1}, \dots, \Lambda_{Si}$, (mit $i = \prod_{i=0}^k n_i$) aber lediglich einzelne Punkte im Lösungsraum betrachtet werden, da die reellen Parameter \vec{r}_0 nicht verändert werden. Der zweite Schritt der Optimierung der reellen Parameter durch die reellen Operatoren X_R und Θ_R erfolgt dann für die gefundene beste Kombination $\vec{d}(k)$. Für diese werden im zweiten Schritt die reellen Parameter $\vec{r}(m)$ in Abhängigkeit der gefundenen nominalen Lösung optimiert.

Im Vergleich zu den folgenden Verfahren führt dieses am schnellsten zu einem Optimierungsergebnis. Mit dem zweistufigen Verfahren wird der Entscheidungsraum und in der Folge der Lösungsraum allerdings erheblich eingeschränkt. Da die Kombination der Maßnahmen nach dem ersten Optimierungslauf gewählt wird und die Suche der besten Kombination auf Basis der reellen Startparameter erfolgt, wird nur die Teilmenge $(\vec{d}(k) \cup \vec{r}(m)) \in (\vec{d} \cup \vec{r})$ des Gesamtlösungsraumes untersucht.

Zu 2. Parallele Optimierung mit allen reellen Parametern: Bei dieser Prozedur werden die nominalen Operatoren X_N, Θ_N und die reellen Operatoren X_R, Θ_R auf dieselben Populationen $P_{i(N)}, P_{i(R)}, P_{i(N)}^*$ und $P_{i(R)}^*$ - genauer gesagt - auf deren nominelle und reelle Anteile angewandt. Im Gegensatz zum vorigen Algorithmus sind die Vektoren \vec{d} und \vec{r} nicht mehr voneinander abhängig. Bei der nominalen Reproduktion und Mutation werden in diesen Verfahren die veränderten reellen Parameter berücksichtigt, da die reellen Mutations- und Reproduktionsprozesse immer auf alle reellen Parameter angewandt werden, unabhängig davon, ob sie bei der gewählten Kombination aktiv sind oder nicht.

Durch dieses Vorgehen ist garantiert, dass der memetische Optimierungsalgorithmus immer die Möglichkeit besitzt, den gesamten Lösungsraum $(\vec{d}_i \cup \vec{r}_i)$ zu berücksichtigen. Das hat aber den Nachteil, dass der Algorithmus mit einer Vielzahl von reellen Parametern

umgehen muss. Die Ergebnisse der Arbeiten von Sharif und Wardlaw (2000), Huebner (2003) und Muschalla (2006) haben aber gezeigt, dass bei steigender Anzahl von reellen Parametern die Optimierung wesentlich mehr Rechenzeit benötigt, und unter Umständen kein befriedigendes Ergebnis ermittelt wird. Die Ergebnisse der kombinatorischen Optimierung in obigen Kapitel 2.7.1 „Problemdomäne Handelsreisendenproblem (TSP)“ untermauern diese Erfahrungen.

Zu 3. Parallele Optimierung mit vorselektierten reellen Parameter: Um die Anzahl der reellen Parameter gering zu halten, wurde beim Algorithmus 3-4 eine zusätzliche Population P_{mem} eingeführt (**Memory** für Gedächtnis). Diese enthält nicht nur die Evaluierungsergebnisse einer Generation P_i sondern speichert die Individuen $\bigcup_{p=0}^{p=\tau} P_{i-p}$ und deren Evaluierungsergebnisse aus den letzten τ Generationen.

Im Gegensatz zum Algorithmus 3-2 und Algorithmus 3-3 erfolgt die Reproduktion und Mutation der reellen Werte nicht mehr für den Gesamtvektor, sondern lediglich für die aktiven reellen Anteile. Dabei kann berücksichtigt werden welchen Vorgänger und welchen Nachfolger die Maßnahme in der Tour hatte. Auf Basis dieser selektierten Lösungen werden dann die reellen Parameter reproduziert und mutiert. Die wesentliche Neuerung dieses Verfahrens ist der Selektionsprozess. Die memetische Selektion $\mathcal{E}_{mem} \bigcup_{p=0}^{p=\tau} P_{i-p}$ betrachtet „lediglich“ den Sublösungsraum einer Maßnahme und vergleicht, welche Qualitätswerte erreicht wurden, wenn diese Maßnahme enthalten war. Nach dem Vergleich werden unter Verwendung dieser besten Lösungen neue reelle Parameter entwickelt. Da Maßnahmen nur mit einer gewissen Wahrscheinlichkeit in jeder Generation vorhanden sind, wird bei dieser Selektion auf den größeren Pool der letzten τ Generationen zurückgegriffen. Wurden für jede Maßnahme die entsprechenden reellen Parameter entwickelt, kann der Gesamtvektor aus $[\vec{d}, \vec{r}]$ evaluiert werden.

Der memetische Selektionsprozess arbeitet mono- als auch multikriteriell. Die Komma-Strategie ist mit dem Operator $\mathcal{E}_{mem,M}$ nicht möglich, da dieser auf mehrere Generationen zurückgreifen muss. Ein Nachteil dieses Verfahren ist sicher das Mischen zweier Lösungspools und deren unterschiedliche Bewertung. Ein Großteil der Lösungen wird nämlich zweifach bewertet. Daher kann es beim Evolutionieren der reellen Parameter der Fall sein, dass auf Lösungen zurückgegriffen wird, die eigentlich aussortiert wurden. Zusätzlich sollte berücksichtigt werden, dass die Programmierung dieses Algorithmus mit nicht un-

erheblichen Aufwand verbunden ist. Außerdem wird die Rechenzeit gerade bei multi-kriteriellen Problemstellungen erheblich erhöht.

Wahl des Verfahrens

In dieser Arbeit wurde das dritte der beschriebenen Verfahren realisiert, da dieses die geringsten Schwächen aufweist. Das erste Verfahren durchforstet lediglich einen Teil des gesamten Lösungsraumes und ist daher nicht in der Lage das globale Optimum zu finden. Das zweite Verfahren durchsucht zwar den Gesamtlösungsraum, wird aber bei steigender Anzahl von Maßnahmen ineffizient, da stets alle reellen Parameter entwickelt werden, obwohl der größte Teil dieser Parameter nicht für die Lösung maßgebend ist. Es wird in Kauf genommen, dass das dritte Verfahren den Gesamtlösungsraum und die Sublösungsräume nicht scharf trennt. Dieses Verfahren berücksichtigt also vorhandene globale Lösungsräume und kleinere, reelle Untermengen innerhalb derer.

3.2.6 Erprobung und Diskussion

Ein detaillierter Vergleich und eine Diskussion wie zum Abschluss der obigen Kapiteln 2.7.1 und 3.1.4 wird für den memetischen Algorithmus nicht vorgenommen, da es für diesen beliebig viele Möglichkeiten der Kombination der Operatoren und Einstellungen gibt, die hier nicht umfassend dokumentiert werden können. Die Erprobung und die Demonstration der Funktionsfähigkeit des Anwendungsbeispiels erfolgen im letzten Kapitel. Dort wird gezeigt, wie mithilfe des hier entwickelten Algorithmus beliebige Kombinationen aus Hochwasserspeichern mit beliebigen Parametern in Lage, Volumen und Abgaben untersucht werden können, ohne die komplexen Interaktionen der Maßnahmen am Gewässer zu vernachlässigen. Die Wechselwirkungen - bedingt durch Abgaben und Volumina der Speicher - werden durch die Evaluierung mit einem Modell berechnet. Im folgenden Kapitel wird das gewählte Simulationsmodell vorgestellt, mit dem der durch die Hochwasserspeicher beeinflusste Abfluss berechnet wird.

4 Modell- und Optimierungssystem

Die oben beschriebenen Algorithmen benötigen einen umfangreichen softwareseitigen Rahmen, in den sie eingebettet werden. Schnittstellen und Steuerungsmechanismen sind erforderlich um Modell und Algorithmen zu kontrollieren. Folgendes Kapitel beschreibt das verwendete Simulationsmodell, den prinzipiellen Ablauf des Modell- und Optimierungssystems und die Steuerungseinheit (Controller) des Optimierungswerkzeuges. Das Optimierungswerkzeug BlueM.Opt und das Simulationsmodell BlueM.Sim sind Teil des am Fachgebiet ihwb entwickelten Softwarepaketes BlueM.

4.1 Das BlueM Softwarepaket

Das Fachgebiet für Ingenieurhydrologie und Wasserbewirtschaftung (ihwb) hat ein Simulations- und Optimierungssoftwarepaket zur Flussgebietsbewirtschaftung entwickelt, welches Methoden und Modelle für eine integrierte Betrachtung ruraler und urbaner Gebiete zur Verfügung stellt. Es handelt sich um die Weiterentwicklung von Modellen und Methoden, die seit vielen Jahren am ihwb verfolgt werden und in diversen Anwendungsprogrammen implementiert wurden. Neben dem hydrologischen Modellkern beinhaltet das BlueM Softwarepaket Werkzeuge zur Optimierung und Autokalibrierung von Modellen, zur Sensitivitätsanalyse von Modellparametern, zur Visualisierung von Optimierungsergebnissen in Form von Scatter Plot Matrizen und zur Visualisierung/Analyse von Zeitreihen (Bach u. a., 2008).

BlueM.Sim ist der hydrologische Rechenkern, der auch in dieser Arbeit für die Simulation verwendet wird. BlueM.Wave dient der Zeitreihenverwaltung und -analyse. BlueM.Opt ist das in dieser Arbeit weiterentwickelte Optimierungswerkzeug. BlueM.Analyser ist ein Echtzeit-Datenmonitor und Auswertungswerkzeug, das via OpenMI angebunden werden kann. BlueM.GIS dient der Generierung der Eingangsdaten für BlueM.Sim auf der Grundlage einer GIS Oberfläche.

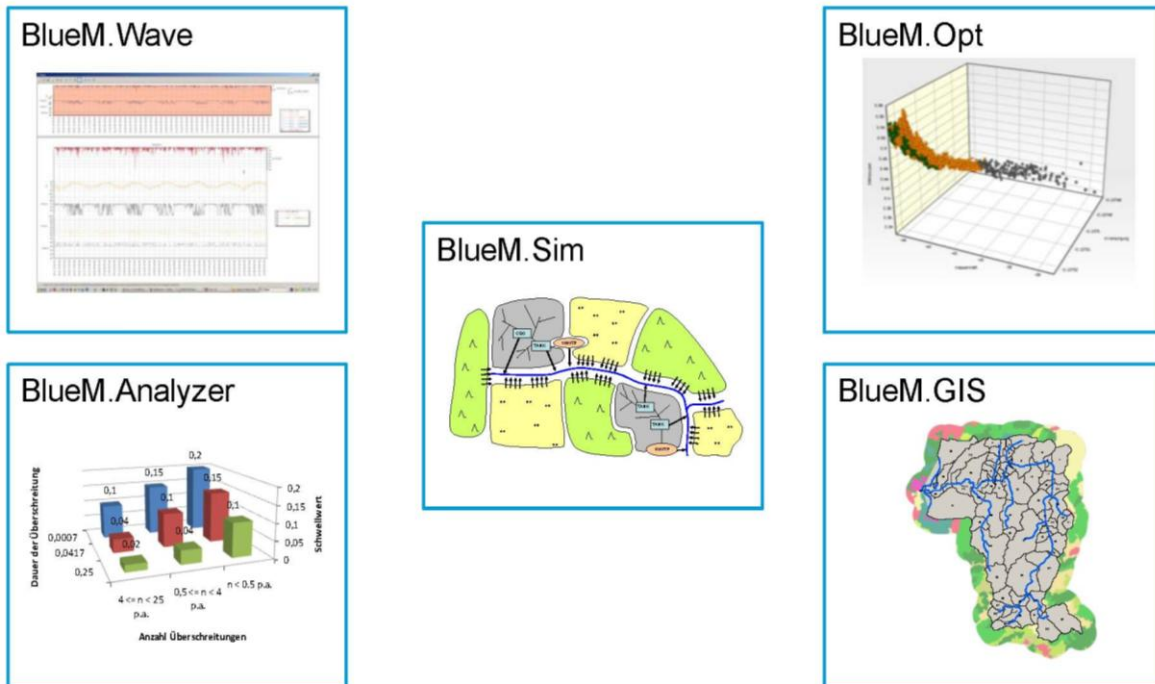


Abbildung 4-1: die Teilpakete des BlueM Simulations- und Optimierungssystems (aus Bach u. a. (2009))

Die wesentlichen BlueM Komponenten sind in Abbildung 4-1 dargestellt. Eine detaillierte Beschreibung der historischen Entwicklung des BlueM Softwarepakets und dessen Funktionalitäten ist Bach u. a. (2009) zu entnehmen.

Bei der hier vorliegenden Kopplung wurde der Simulationskern BlueM.Sim mit dem Optimierungswerkzeug BlueM.Opt gekoppelt. Im folgenden Kapitel wird das Simulationsmodell kurz beschrieben.

4.2 BlueM.Sim

Der Rechenkern BlueM.Sim ist eine in FORTRAN programmierte Simulationssoftware zur konzeptionellen Modellierung ruraler und urbaner Einzugsgebiete. Es enthält u. a. ein komplexes Steuerungsmodul, mit dem sich Talsperren und andere Bauwerke in Abhängigkeit von beliebigen Systemzuständen regeln lassen. Dies ermöglicht auch die Definition von Hochwasserrückhaltebecken und erlaubt es, Betriebspläne von Talsperren und Hochwasserrückhaltebecken zu berücksichtigen. Durch die Abbildung des hydrologischen Modells in einer hierarchischen Softwarearchitektur kann die Erweiterungen um weitere Systemelemente relativ leicht erfolgen (Bach u. a., 2008).

Das Modell enthält derzeit die in Tabelle 4-1 aufgelisteten Systemelemente.

Tabelle 4-1: Systemelemente des BlueM.Sim Simulationsmodells (Ostrowski et al., 2002; 2009)

Element	Wichtige Belastungen	Eigenschaften	Elementausgang
Natürliches Einzugsgebiet	<ul style="list-style-type: none"> – Niederschlag – Temperatur – Verdunstung 	<ul style="list-style-type: none"> – Bodenkennwerte – Abflussbildung – Abflussaufteilung – Abflusskonzentration – ... 	<ul style="list-style-type: none"> – Oberflächenabfluss – Basisabfluss – Gesamtabfluss – ...
Urbanes Einzugsgebiet	<ul style="list-style-type: none"> – Niederschlag – Einwohner / Gewerbe 	<ul style="list-style-type: none"> – Anteil befestigter Flächen – Abflussbildung – Abflussaufteilung – Abflusskonzentration – ... 	<ul style="list-style-type: none"> – Regenwetterabfluss – Trockenwetterabfluss – Gesamtabfluss – ...
Einleitung		<ul style="list-style-type: none"> – Wassereintrag in das System 	<ul style="list-style-type: none"> – Abfluss
Transportstrecke	<ul style="list-style-type: none"> – Zufluss 	<ul style="list-style-type: none"> – Translation – Retention 	<ul style="list-style-type: none"> – Abfluss
Verbraucher	<ul style="list-style-type: none"> – Zufluss 	<ul style="list-style-type: none"> – Verbrauchsverhalten – Zuschuss aus anderen Gebieten – Wiedereinleitung in das System 	<ul style="list-style-type: none"> – Wiedereinleitung – Zuschuss – Gesamtabfluss
Verzweigungen	<ul style="list-style-type: none"> – Zufluss 	<ul style="list-style-type: none"> – Verteilungsvorschrift 	<ul style="list-style-type: none"> – Zwei Abflüsse
Speicher – Talsperren – HRB – RRB	<ul style="list-style-type: none"> – Zufluss optional: – Niederschlag – Verdunstung 	<ul style="list-style-type: none"> – Speicherinhaltskurve – Speicheroberflächenkurve – Leistungsfähigkeit der Betriebseinrichtungen – Betriebsregeln – (Versickerungsverhalten) – ... 	<ul style="list-style-type: none"> – Abgaben – Speicherinhalt – Wasserstand
Regenüberläufe	<ul style="list-style-type: none"> – Zufluss 	<ul style="list-style-type: none"> – Verteilungsvorschrift 	<ul style="list-style-type: none"> – Zwei Abflüsse
Regenüberlaufbecken	<ul style="list-style-type: none"> – Zufluss 	<ul style="list-style-type: none"> – Speicherinhaltsangaben 	<ul style="list-style-type: none"> – Zwei Abflüsse

Detaillierte Beschreibungen des verwendeten Modells sind in den Arbeiten H. Lohr (2001), Ostrowski et al. (2002), Klawitter (2006), Muschalla (2006) und Bach u. a. (2009) zu finden. Zur Berechnung der Speicher und einiger anderer Systemelemente wird der universelle, nichtlineare Speicherbaustein aus Ostrowski (1992), Mehler und Ostrowski (1998) und Ostrowski et al. (1999) eingesetzt.

4.3 Das kombinierte Modell und Optimierungssystem

Wie bereits erwähnt, besteht ein wesentlicher Vorteil evolutionärer Algorithmen in der Trennung zwischen Modellsystem und Optimierungssystem. Für die Optimierung ist nur ein geringer Informationsaustausch zwischen Modell und Algorithmus erforderlich. Der Austausch erfolgt jeweils vor und nach der Evaluierung durch das Simulationsmodell. Abbildung 4-2 zeigt, dass vor der Evaluierung die vom Optimierungsalgorithmus entwickelten Genotypen (in Form von ASCII Dateien) an das Simulationsmodell übergeben werden. Der übergebene Genotyp \vec{x} besteht lediglich aus den erforderlichen Simulationsparametern. Das Simulationsmodell evaluiert f_{Mi} diese Parameter und übergibt die Ergebnisse an den Optimierungsalgorithmus (als ASCII Datei) zurück. Der Algorithmus wertet die Ergebnisse hinsichtlich der ihm bekannten Zielfunktion aus und berechnet den oder die Qualitätswerte $F(\vec{x})$. Mithilfe der Zielfunktion können mehrere Ergebnisse aggregiert oder funktionale Zusammenhänge hergestellt werden. Die ermittelten Qualitätswerte bilden den Phänotyp des evaluierten Genotyps und dienen der Selektion, Reproduktion und Mutation der nächsten Eltern.

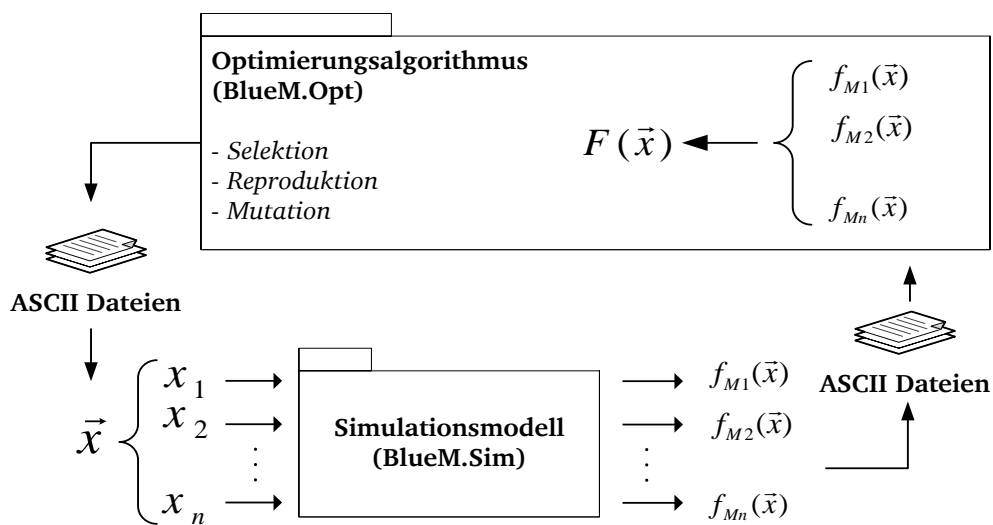


Abbildung 4-2: Modell und Optimierungssystem (angelehnt an Hättenschwiler und Gachet (2009))

Die Anbindung der Simulationsmodelle als auch der Optimierungsalgorithmen wurde generalisiert, um weitere geeignete Simulations- und Optimierungskerne einbinden zu können. Hierfür wurde ein Controller entwickelt, welcher den Datenaustausch zwischen Modell und Algorithmus steuert. Beide Modell und Algorithmus sind passiv angebunden. Das bedeutet, dass der Controller die erforderlichen Strukturen enthält, die für das Abarbeiten der Generations- und Nachfahenschleifen innerhalb der Prozesse Mutation, Reproduktion und Selektion der Optimierungsalgorithmus, sowie zum Starten der Simulation notwendig sind. Er ist ebenso für das Übertragen der Informationen von und zum Simulationsmodell verantwortlich.

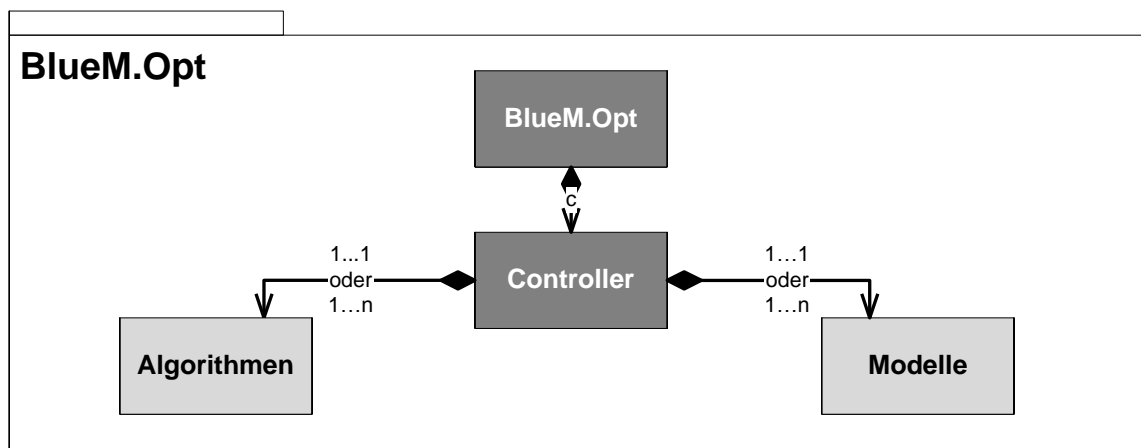


Abbildung 4-3: Steuerung des Optimierungsalgorithmus und des Simulationsmodells mithilfe des Controllers

Obige Abbildung 4-3 zeigt die Abhängigkeit der Algorithmen und Modelle vom Controller. Standardmäßig wird an den Controller *ein* Algorithmus angebunden. Dieses Verfahren wurde auch für die in Kapitel 3.1 beschriebene nominell diskrete Optimierung eingesetzt. Für den memetischen Algorithmus aus Kapitel 3.2 wurde eine 1 zu n Anbindung entwickelt (siehe Abbildung 4-4). Zu diesem Zweck wurden alle Operatoren des reellwertigen Algorithmus aus der Arbeit von Muschalla (2006) und des nominell diskreten Algorithmus generalisiert, um sie nach Bedarf ansprechen zu können. Der memetische Algorithmus benötigt daher keine neuen Operatoren, er verwendet die vorhandenen und führt mit obig beschriebener Prozedur (Algorithmus 3-4) den zusätzlichen Individuenspeicher P_{mem} ein. Die Generalisierung und die Mehrfachanbindungen ermöglichen dieses Vorgehen.

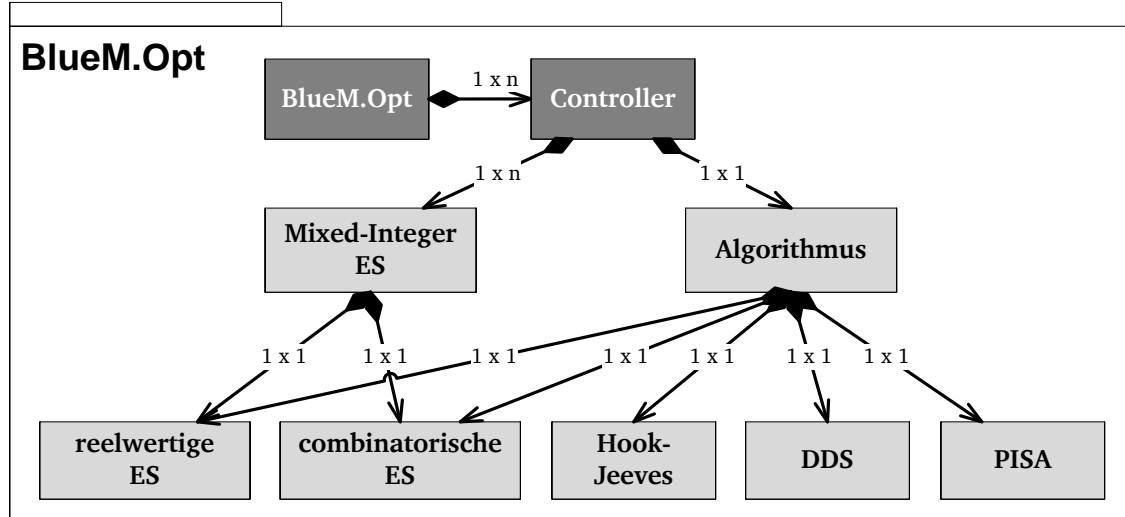


Abbildung 4-4: Anbindung der Optimierungsalgorithmen an den Controller. Der memetische Algorithmus setzt Operatoren des kombinatorischen und des reellwertigen Algorithmus ein.

Das Simulationsmodell wird ebenfalls mit 1:1 Verbindung an den Controller angebunden (Abbildung 4-5). Auch hier wurden zwischenzeitlich Schnittstellen für weitere Modelle geschaffen. Jedes Modell besitzt einen eigenen Adapter. Falls das Modell es softwareseitig erlaubt mehrere Instanzen gleichzeitig auszuführen, ist es möglich, Simulationen parallel auszuführen. Auf diese Weise kann der Optimierungsprozess um ein Vielfaches beschleunigt werden.

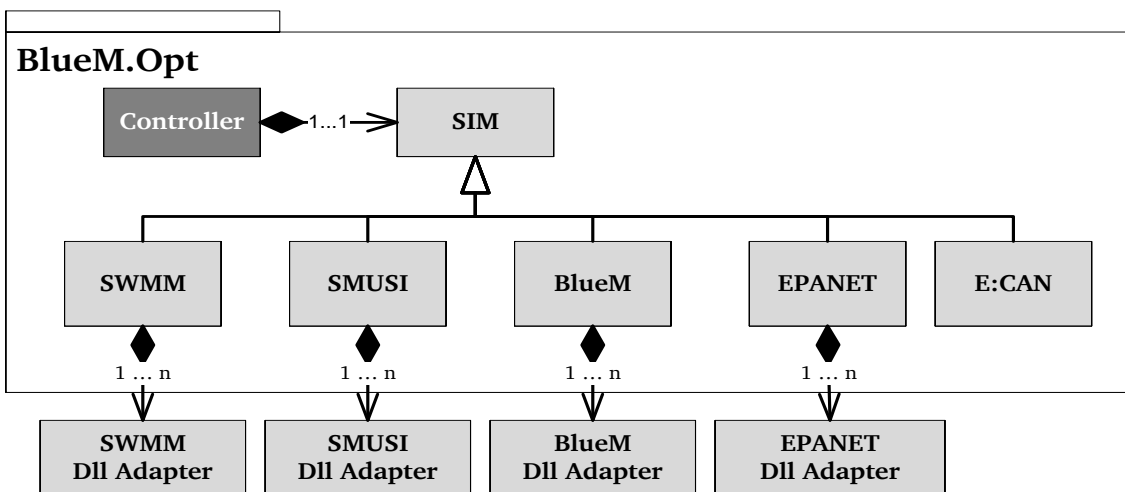


Abbildung 4-5: Anbindung der Simulationsmodelle

Das Kapitel 5 geht detailliert auf die Methoden und die Realisierung der Parallelisierung ein.

4.4 Modellierung des Optimierungsdatensatzes

Der zur Optimierung der Hochwasserspeicher erstellte Datensatz besteht im Wesentlichen aus den Systemelementen Einleitung, Transportstrecke, Verbraucher, Verzweigungen und Speicher. Eingangsinformationen sind gemessene Abflusszeitreihen.

Eine besondere Rolle bei der Optimierung spielen die Elemente Speicher, Transportstrecke und Verzweigung. Mittels Speichern und Transportstrecken werden technische Maßnahmen wie Hochwasserrückhaltebecken, Polder, Talsperren, Sohlvertiefungen und Gerinneaufweitungen abgebildet und deren Parameter und Steuerstrategien durch die Evolutionsstrategie optimiert. Mit dem Systemelement Verzweigung wurde die Variantenoptimierung ermöglicht.

Um die vom Optimierungsalgorithmus gewählten Maßnahmen zu aktivieren, ist ein „Schaltplan“ erforderlich, der die Repräsentation auf das Optimierungsmodellsystem schaltet.

Tabelle 4-2: Schematischer Schaltplan für die kombinatorische Steuerung des Modelldatensatzes

		Schaltung (Achtung Leerzeichen)						
Position	Maßnahme	VerZw 1	VerZw 2	VerZw 3				
Standort A	Maßnahme 0	VZ_1 0	X X	X X				
	Maßnahme 1	VZ_1 1	VZ_2 0	X X				
	Maßnahme 2	VZ_1 1	VZ_2 1	X X				
Standort B	Maßnahme 0	VZ_3 0	X X	X X				
	Maßnahme 1	VZ_3 1	X X	X X				
Standort C	Maßnahme 0	VZ_4 0	VZ_5 0	X X				
	Maßnahme 1	VZ_4 1	VZ_5 1	X X				
	Maßnahme 2	VZ_4 0	VZ_6 0	X X				
	Maßnahme 3	VZ_4 1	VZ_6 1	VZ_7 0				
	Maßnahme 4	VZ_4 1	VZ_6 1	VZ_7 1				

In Tabelle 4-2 ist schematisch ein Schaltplan für die Kombinatorik dargestellt. In der Ersten Spalte steht der Name des Standortes, rechts daneben die zugehörigen Maßnahmen für diesen Standort. Die übrigen Spalten enthalten die Namen der Verzweigungen, mit Hilfe derer die Abflüsse zur den entsprechenden Maßnahmen geleitet werden. Jede Verzweigung hat einen Zufluss und zwei Abflüsse. Mit 0 und 1 wird definiert, zu welchem Abfluss die Hochwasserwelle geleitet wird. Damit können zwei Maßnahmen durch eine Verzweigung gesteuert werden. Sind mehr als zwei Maßnahmen an einem Standort platziert, werden entsprechend mehr Verzweigungen benötigt.

Die Abbildung der Kombinatorik wäre theoretisch auch mit dem im Simulationsmodell verwendeten Systemplan möglich. Dort ist definiert welche Vorgänger und Nachfolger ein

Systemelement besitzt. Allerdings müssten Systemelemente die in der aktuellen Kombination des Nachfahren nicht aktiviert sind für dessen Evaluierung gänzlich aus dem Datensatz gelöscht werden. Da Informationen über Systemelemente über mehrere Dateien und teilweise redundant verteilt sind, ist die Steuerung der Kombinatorik über den Systemplan nicht praktikabel.

Bei der Modellierung muss wie bereits erwähnt darauf geachtet werden, die Zahl der Optimierungsparameter möglichst gering zu halten. Die Zahl der Optimierungsparameter erhöht die Rechenzeit erheblich. Darüber hinaus sollte auch zunächst die Modellkomplexität gering gehalten werden. Statt einer Maßnahme an einer potenziellen Position werden zusätzliche alternative Maßnahmen platziert. Allerdings können bei einer Vielzahl von Kombinationsmöglichkeiten nicht alle validiert werden. Wird ein Speicher in dem hier verwendeten Simulationsmodell mit einem sehr viel zu großen Volumen $V \gg V_{max}$ beschickt, wird die Simulation abgebrochen. Dies gilt es durch eine „angemessene“ Modellkomplexität zu verhindern. Die Komplexität und die im folgenden Kapitel beschriebene Performanz müssen der anvisierten Modellierung und Optimierung angemessen sein.

4.5 Performanz des Modell- und Optimierungskonzeptes

Zur Ermittlung von dicht besetzten Paretofronten einer multikriteriellen Optimierung sind häufig Tausende von Simulationen erforderlich. Unter Verwendung aktueller, allgemein verfügbarer Personal Computer können so Optimierungszeiten von mehreren Stunden oder Tagen entstehen. Der rechenintensive Teil bei der Kombination von Optimierungsalgorithmen und Simulationsmodellen besteht meist allerdings nicht aus den Prozessen, welche für die Optimierung erforderlich sind, sondern aus der Evaluierung der Parameter mit dem Simulationsmodell. Die Evaluierung erfordert je nach Problemstellung 90 % bis 98 % der gesamten Optimierungszeit. Soll die Optimierungszeit reduziert werden, muss folglich zunächst die erforderliche Rechenzeit für die Evaluierung reduziert werden. In dieser Arbeit wird zur Evaluierung des Speichersystems ein hydrologisches Simulationsmodell eingesetzt. Dementsprechend wird zur Auswertung der Zielfunktionen der Abfluss verwendet. Für eine detaillierte Betrachtung des Wasserstands wäre eine hydraulische Modellierung angemessen. Darauf muss allerdings verzichtet werden, da hydraulische Modelle erheblich mehr Rechenzeit benötigen. Dies beschränkt zum einen die Genauigkeit der Wellenablaufberechnung und die Art an Maßnahmen, die bei der Optimierung berück-

sichtigt werden können, da beispielsweise Gerinneaufweitungen weniger exakt berechnet werden können.

4.6 Performanz des Modelldatensatzes

Hierfür sind mehrere Ansätze denkbar. Die Simulationszeit kann seitens der Modellierung reduziert werden, durch:

- **Reduktion der zeitlichen Diskretisierung.** Der Simulationszeitraum sollte so kurz wie möglich und die Zeitschrittlänge so lange wie möglich gewählt werden. Beide aber hinreichend, um den Anforderungen der Problemstellung zu genügen. Zusätzlich kann durch die Anwendung eines variablen Zeitschritts eine bessere Erfassung des Spitzenabflusses erreicht und gleichzeitig die Vor- und Nachperiode eines Ereignisses erfasst werden (Ostrowski und Wolf, 1984).
- **Reduktion der Modellkomplexität.** Das Modell sollte nur die Eigenschaften des wasserwirtschaftlichen Systems erfassen, die für die Lösung der jeweils gestellten Aufgaben von Bedeutung sind. Allerdings müssen die Anforderungen an die problemadäquate Diskretisierung gewahrt bleiben.
- **Reduktion der räumlichen Diskretisierung.** Der Umfang des modellierten Raumes sollte so klein wie möglich und die räumliche Diskretisierung sollte so groß wie möglich gewählt werden. Hierbei kann es sinnvoll sein, für unterschiedliche Teilprozesse unterschiedliche Diskretisierungen zu wählen. Unterschiedliche räumliche Diskretisierungen erfordert die adäquate Strukturierung des Modells um homogene Systemantworten zu erhalten.

Falls der Quellcode des verwendeten Modells verfügbar ist, gibt es Möglichkeiten durch eine angepasste Programmierung die Simulationszeit zu verringern. Das Potenzial hierbei ist allerdings wesentlich von der verwendeten Programmiersprache und dem verwendeten Compiler abhängig. Die Effizienz von Programmiersprachen und Programmierparadigmen ist ein weites Feld und soll hier nur teilweise erschlossen werden. Näheres zur effizienten Programmierung ist in Louden (2002) zu finden. Grundsätzlich ist auf das Zusammenspiel zwischen der verwendeten Sprache und der Problemstellung sowie dem Compiler und der für die Simulation verwendeten CPU zu achten.

5 Parallelisierung des Modell- und Optimierungssystems

Das am Fachgebiet für Ingenieurhydrologie und Wasserbewirtschaftung (ihwb) der TU-Darmstadt entwickelte BlueM.Sim ist in der Programmiersprache FORTRAN geschrieben. Fortran wurde 1958 veröffentlicht und gilt als die erste „höhere“ Programmiersprache (Brockhaus, 2009). Selbst im Vergleich zu neuen Programmiersprachen zeichnet sich Fortran vor allem durch seine schnelle Laufzeit aus. Es wird noch heute weiterentwickelt und ist in wissenschaftlichen Anwendungen weit verbreitet. Für die Compilierung des BlueM.Sim Codes wird am Institut der Intel® Fortran Compiler verwendet (Bach u. a., 2009). Die Implementierung des Optimierungsalgorithmus wurde in der Arbeit von Huebner (2003) innerhalb der Excel Visual Basic for Applications Umgebung begonnen (Kofler, 2000; Ravens, 2004). Muschalla (2006) hat den Code nach Visual Basic 6 migriert und entwickelte die wesentliche Grundlage für den heutigen im .NET Framework fortgeführten evolutionstrategischen Optimierungsalgorithmus.

Es folgt eine Auswahl von Methoden, die sich sowohl auf die Effizienz der Programmierung, als auch auf Möglichkeiten zur Parallelisierung beziehen. Die Steigerung der Rechenzeit durch den Einsatz neuester Hardwaretechnologie, wie beispielsweise Prozessoren, Speicher oder Netzwerktechnik wurde hier ebenso wenig berücksichtigt, wie das Grid-Computing, da es als eine logische, ortsungebundene Weiterentwicklung des Cluster-Computing angesehen werden kann (Bengel u. a., 2008). Die Optimierungszeit kann auf die folgende Weise reduziert werden:

- 1. Anpassung von rechenzeitintensiven Prozessen, Funktionen und Befehlen.**

Der Code des verwendeten Modells muss nach rechenzeitintensiven Operationen untersucht werden. Als zeitintensiv identifizierte Passagen, Funktionen oder Befehle, zum Beispiel Stringoperationen, können dann gegebenenfalls hinsichtlich ihrer Ausführungszeit verbessert werden.

- 2. Manuelle Parallelisierung durch Open specifications for Multi Processing**

(**OpenMP**) ist eine ist eine Programmierschnittstelle für Fortran und C/C++, die zum Zwecke der parallelen Programmierung für Mehrprozessor-Systeme entwickelt wurde (Chandra u. a., 2000). Dabei können zwei oder mehr Prozesse auf den gleichen Adressraum im Hauptspeicher zugreifen.

-
3. **Automatische Parallelisierung und Optimierung durch High-Performance Parallel Optimization (HPO)** restrukturiert und optimiert Schleifen, um sicherzustellen, dass die automatische Vektorisierung, OpenMP oder die Auto-parallelisierung den schnellen Zwischenspeicher des Prozessors, die Hauptspeicherzugriffe und Mehrkernprozessoren bestmöglich verwendet werden. Der Vollständigkeit halber seien hier **Interprocedural Optimization (IPO)** und **Profile-Guided Optimization (PGO)** genannt. Die erste Methode erhöht die Performanz von Funktionen, die Zweite verringert und reorganisiert den Quelltext (Intel Corporation, 2009a).
 4. **Manuelle Parallelisierung durch das parallele Ausführen zweier oder mehr Simulationen auf einem Mehrkernprozessor.** Bei der Optimierung wird das Simulationsmodell vollständig vom Optimierungsalgorithmus kontrolliert. Daher ist es möglich, auf einem Mehrkernprozessor zwei oder mehr Simulationskerne parallel auszuführen. Dabei wird jedem Simulationskern ein eigener Simulationsordner und ein eigener Thread zugewiesen. Als Threads werden solche sich in der Ausführung befindende Programmstücke bezeichnet, die zwar eigene Register, Stack und Programmzähler haben, andere Informationen aber mit anderen Threads teilen (Rauber und Rüniger, 2000). Dadurch wird dem Betriebssystem ermöglicht, die Threads auf die verschiedenen physikalischen oder virtuellen Kerne des Prozessors zu verteilen.
 5. **Manuelle Parallelisierung durch das parallele Ausführen zweier oder mehr Simulationen in einem Netzwerkcluster (Cluster Computing).** In diesem Fall werden die Simulationskerne nicht auf die verschiedenen Kerne eines Prozessors verteilt, sondern auf verschiedene Rechner in einem Netzwerk. Dafür ist es erforderlich, auf jedem beteiligten Rechner das Simulationsprogramm vorzuhalten. Der Optimierungsalgorithmus oder gegebenenfalls ein gesonderter „Kontroller“ verteilt dann die zu evaluierenden Simulationsdatensätze auf die verschiedenen verfügbaren Rechner. Nach Beendigung der Simulation erhält der Optimierungsalgorithmus die Simulationsergebnisse. Diese Art der Performanzverbesserung wurde am Institut erfolgreich in den Arbeiten von Muschalla (2006) und Kerber (2009) erprobt.

Weitere Methoden zur effizienten Programmierung sind in Louden (1998), sowie in Pratt und Zelkowitz (1999) zu finden.

5.1 Methodendiskussion

Die unter Punkt 1. beschriebene Methode „**Suche nach rechenzeitintensiven Prozessen, Funktionen und Befehlen**“ erfordert eine gründliche Untersuchung des Codes. Hierfür sind allerdings spezielle Werkzeuge zur Performance Analyse erforderlich, wie beispielsweise VTune von Intel Corporation (2009b), AMD CodeAnalyst von AMD, Inc. (2009) oder Linux Trace Toolkit von Opsys, Inc. (2009). Je nach Qualität des ursprünglichen Quellcodes kann der Gewinn hinsichtlich der Ausführungsgeschwindigkeit erheblich sein. Allerdings ist dieser Gewinn nur mit großem Aufwand erzielbar und nicht beliebig zu steigern.

Alle weiteren oben genannten Methoden beziehen sich hauptsächlich auf die Parallelisierung des auszuführenden Quelltextes oder von Teilen des Quelltextes. Hierbei spielt das *Amdahlsche Gesetz* (Amdahl, 1967) im Modell der Informatik aber auch der Betriebswirtschaftslehre eine wesentliche Rolle. Das Gesetz - benannt nach dem Computerarchitekten Gene Amdahl - beschreibt, dass die Beschleunigung von Programmen durch parallele Ausführung des Quelltextes, hauptsächlich durch den sequenziellen Anteil einer Problemstellung beschränkt ist (Rauber und Rüniger, 2000). Es wird dazu verwendet im Bereich des parallelen Rechnens den theoretisch maximal erreichbaren Geschwindigkeitsgewinn, bei p Prozessoren und in den im Programm vorhandenen sequenziellen, nicht parallelisierbaren Anteilen f sowie den parallelisierbaren Anteilen $(1 - f)$ zu ermitteln (Bengel u. a., 2008).

Die Zeit zur reinen sequenziellen Ausführung des gesamten Quelltextes sei $T'(n)$, wobei n die zu lösende Problemgröße beschreibt. Die Gesamtlaufzeit des Quelltextes mit zwei oder mehr Prozessoren $T_p(n)$ ist dann größer als die Summe aus dem sequenziellen Anteil und dem parallelen Anteil, dividiert durch die Anzahl der zur Verfügung stehenden Prozessoren:

$$T_p(n) \geq f \cdot T'(n) + \frac{(1 - f) \cdot T'(n)}{p} \quad (5-1)$$

Der maximal mögliche Geschwindigkeitsgewinn ist demzufolge:

$$S_p(n) = \frac{T'(n)}{T_p(n)} \quad (5-2)$$

$$= \frac{T'(n)}{f \cdot T'(n) + \frac{(1-f) \cdot T'(n)}{p}} \quad (5-3)$$

Amdahlsches Gesetz:

$$= \frac{1}{f + \frac{(1-f)}{p}} \quad (5-4)$$

- Mit:
- f : *Sequenzieller Anteil des Algorithmus*
 - $(1-f)$: *Parallelisierbarer Anteil des Algorithmus*
 - n : *Zu lösende Problemgröße*
 - $T'(n)$: *Zeit zur sequenziellen Ausführung*
 - $T_p(n)$: *Zeit zur parallelen Ausführung*
 - $S_p(n)$: *Potenzieller Geschwindigkeitsgewinn*
 - p : *Anzahl der Prozessoren*

Für $f > 0$ und eine große Anzahl von Prozessoren $p \rightarrow \infty$ folgt aus dem Amdahlschen Gesetz näherungsweise:

$$S_p(n) \leq \frac{1}{f} \quad (5-5)$$

Aus dieser Gleichung geht hervor, dass bereits ein kleiner nicht parallelisierbarer Anteil des Algorithmus die maximal mögliche Beschleunigung begrenzt.

Unter der Annahme, dass der sequenzielle Anteil f beträgt 10 %, dann kann die Lösung des Problems maximal um den Faktor 10 beschleunigt werden, unabhängig davon, wie groß die Problemstellung n ist. Aus diesem Grund sind parallele Berechnungen nur für eine kleine Anzahl von Prozessoren oder nur mit sehr kleinen Werten von f sinnvoll. Des Weiteren würde sich der Einsatz von sog. Massenparallelrechnern (MPR), mit mehreren Hunderten oder gar Tausenden Prozessoren nicht lohnen. Abbildung 5-1 zeigt den maximalen Geschwindigkeitsgewinn nach dem Amdahlschen Gesetz in Abhängigkeit der verfügbaren Prozessoren und des parallelisierbaren Programmanteils.

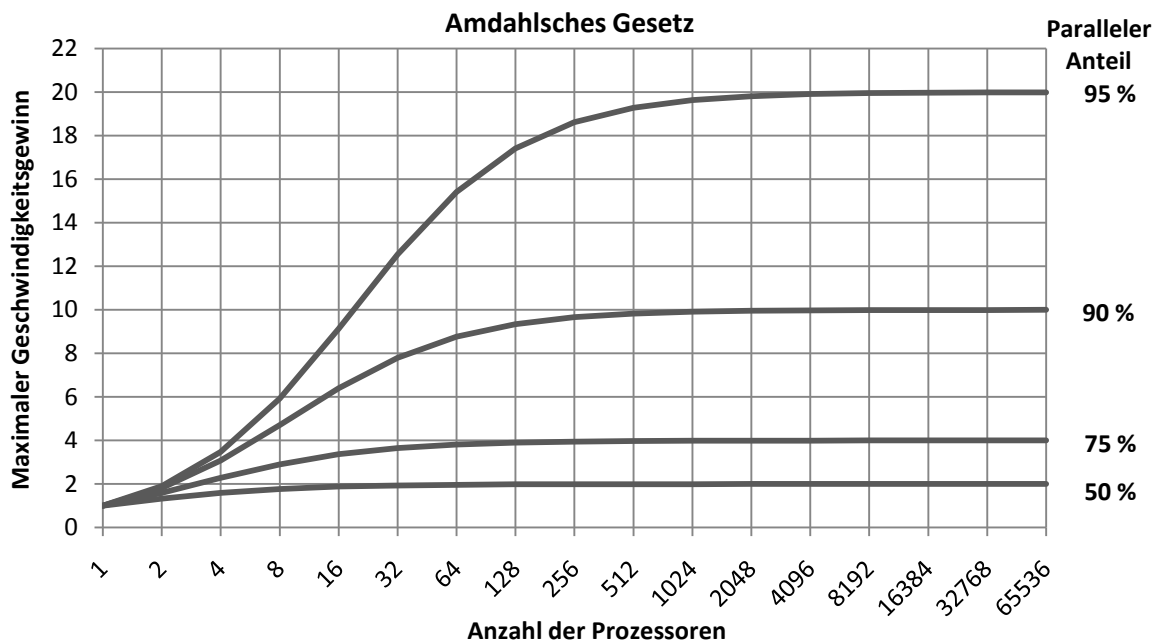


Abbildung 5-1: Geschwindigkeitsgewinn bei der Bearbeitung eines parallelen Problems unter Verwendung von Mehrkernprozessoren bei verschiedenen großen parallelen Anteilen

Für die unter **Punkt 2** („Manuelle Parallelisierung durch OpenMP“) und **Punkt 3** („Automatische Parallelisierung und Optimierung HPO“) aufgeführten Methoden ist der parallel ausführbare Teil nur mit erheblichem Aufwand exakt zu bestimmen. Sequenzielle Programmteile sind, zum einen Seitens der Programmierung, zur Prozessinitialisierung oder Speicherallokation erforderlich. Aber auch seitens der Problemstellung sind sie notwendig, da häufig neue Ergebnisse auf früheren Ergebnissen beruhen und iterative berechnet werden müssen. Bei der hier vorliegenden Problemstellung ist die hydrologische Berechnung vor allem die Berechnung des Gerinnes ein sequenzieller, nicht parallelisierbarer Prozess. Parallel könnten die einzelnen unabhängigen hydrologischen Einheiten oder gegebenenfalls die Seitengerinne berechnet werden. Erschwerend für die Parallelisierung kommt hinzu, dass der vorhandene Fortran Code historisch gewachsen ist. So war es beispielsweise in der Vergangenheit üblich, teilweise einzig möglich, sogenannte Sprunganweisungen (*goto*) statt Schleifen zu verwenden. Daher enthält das BlueM noch heute viele *goto* Anweisungen. Für die automatische als auch für die manuelle Parallelisierung mittels OpenMP ist es erforderlich, dass weder *goto* noch sog. *break* Anweisungen vom Inneren der Schleife nach Außen erfolgen. Auch Ausnahmebehandlungen (*exceptions*) müssen innerhalb der Schleife behandelt werden. Dieser Umstand ist nur mit

erheblicher Umstrukturierung des Codes und dadurch bedingter neuer Fehlerquellen, zu erfüllen.

Unter der Annahme, dass der parallelisierbare Teil bei 60 % liegt, wäre nach dem Amdahlschen Gesetz (5-5) ein maximaler Geschwindigkeitsgewinn von 150 % möglich. Daher ist es vielversprechender, die Parallelisierung in der Kombination aus Simulationsmodell und Optimierungsalgorithmus zu suchen.

Innerhalb des Optimierungsprozesses werden der zu simulierende Modelldatensatz, die Parameter, der Simulationszeitraum etc. vollständig vom Optimierungsalgorithmus kontrolliert. Falls Modelldatensätze und zugehörige Parameter seitens des Optimierungsprozesses jeweils für eine Generation rekombiniert und mutiert werden, stehen alle Informationen zur Verfügung, um die notwendigen Evaluierungen durch das Simulationsmodell für eine Generation parallel auszuführen. Innerhalb des neu entwickelten nominell diskreten Optimierungsalgorithmus wurde die nächste Generation bereits „als Ganzes“ generiert und dann „lediglich“ abgearbeitet. Der von Muschalla (2006) entwickelte Algorithmus arbeitete noch sequenziell, sprich ein Nachfahre wurde generiert und in der Folge auch evaluiert. Um mehrere Nachfahren parallel abzuarbeiten, wurde dieser Algorithmus entsprechend modifiziert, sodass jeweils eine vollständige Generation zur Evaluierung zur Verfügung steht.

Der sequenzielle Anteil des Optimierungsprozesses besteht im Wesentlichen aus der Initialisierung des Gesamtsystems und aus dem Generieren und Verarbeiten einer gesamten Generation. Gerade bei monokriteriellen Problemstellungen laufen diese Prozesse sehr schnell ab, da das beste Evaluierungsergebnis eindeutig identifizierbar ist bzw. das rechenzeitintensive „Non-Dominated-Sorting“ entfällt (Chuan Shi u. a., 2005; Du u. a., 2007; Wong, 2009). Der parallelisierbare Teil, die Simulation des Datensatzes, benötigt wesentlich mehr Rechenzeit. In dem hier vorliegenden Fall wird der Anteil des parallelisierbaren Teils auf $(1 - f) \sim 95\%$ geschätzt. Hierbei handelt es sich allerdings um die Simulation einzelner, relativ kurzer Hochwasserereignisse. In den meisten Fällen wie beispielsweise der Kalibrierung von Niederschlags-Abfluss-Modellen sind die Simulationszeiten wesentlich höher, sodass von einem parallelisierbaren Anteil von $(1 - f) \geq 95\%$ ausgegangen werden kann.

Für den hier vorliegenden Fall, der Kombination eines Simulationsmodells und eines evolutionstheoretischen Optimierungsalgorithmus, wird der parallelisierbare Anteil konservativ auf 95 % geschätzt. Dem Amdahlschen Gesetz folgend liegt der maximale Geschwindigkeitsgewinn bei einem Faktor von 20 – unter der Voraussetzung, dass entsprechend viele Prozessoren zur Verfügung stehen.

Die Verteilung der Simulation auf mehrere Prozessoren sehen die Fälle, unter **Punkt 4** („Manuelle Parallelisierung durch das parallele Ausführen zweier oder mehr Simulationen auf einem Mehrkernprozessor“) und **Punkt 5** („Manuelle Parallelisierung durch das parallele Ausführen zweier oder mehr Simulationen in einem Netzwerkcluster“) vor. Diese beiden Methoden unterscheiden sich lediglich darin, dass im einen Fall – falls vorhanden – mehrere Kerne eines Prozessors genutzt werden und im anderen Fall mehrere Prozessoren auf verschiedenen Rechnern in einem Netzwerk (Cluster) genutzt werden.

Letzteres wurde in den Arbeiten von Muschalla (2006) und Kerber (2009) erfolgreich realisiert. Pfister (1998) definiert ein Cluster als eine Art paralleles System, welches aus miteinander verbundenen, eigenständigen Computern besteht und als eine einzige, verschmolzene Computereinheit verwendet wird. Von einem Cluster im engeren Sinne wird meist gesprochen, wenn sich die Knoten (auch Clients oder Slaves) unter der Kontrolle eines Masters befinden. Diese Master Slave Architektur finden auch in den Arbeiten von Muschalla (2006) und Kerber (2009) Anwendung.

Der Einsatz eines Clusters birgt einige Nachteile. Beispielsweise muss die Simulationssoftware auf die einzelnen Knoten verteilt werden. Da die Knoten ggf. aus einzelnen PCs bestehen, müssen diese auch administriert werden. Bei einem heterogenen Grid mit unterschiedlichen Prozessoren ist zusätzlich zu berücksichtigen, dass ein Lastverteilungsverfahren eingesetzt wird, um die verschiedenen Rechenzeiten auszugleichen und Leerlaufzeiten zu minimieren (Alex und Ganten, 2007; Bengel et al., 2008). Je nach Problemstellung kann ein weiteres Nadelöhr das verwendete Netzwerk des Clusters sein. Zwar werden in der Informatik Hochgeschwindigkeitsnetzwerke verwendet, aber selbst diese sind um Größenordnungen langsamer als die Verbindungen zwischen Prozessor und Hauptspeicher von Parallelrechnern. Daher ist bei einem Cluster zu berücksichtigen, dass der Datentransfer über das Netzwerk möglichst gering ist (Bauke und Mertens, 2006).

Von Vorteil ist die Skalierbarkeit eines heterogenen Clusters aus einzelnen PCs. Soll beispielsweise die Leistung des Clusters erhöht werden, müssen dem Netzwerk lediglich weitere PCs hinzugefügt werden. So können nachts nicht verwendete Arbeitsrechner zusätzlich zur Leistungserhöhung eingesetzt werden. Ebenso können die Kosten gering gehalten werden, da die Infrastruktur in Form eines Netzwerkes und die nötige Hardware in Form der Desktopcomputer meist bereits vorhanden sind. Außerdem schließt paralleles Rechnen in einem Cluster gleichzeitiges paralleles Rechnen auf Mehrkernprozessoren nicht aus.

Im Vergleich zur Clusterlösung birgt die Verteilung der Rechenlast auf mehrere Prozessoren oder Mehrkernprozessoren einige Vor- und Nachteile. Vorteilhaft wirken sich die folgenden Punkte aus:

- Der zur Optimierung verwendete Rechner muss nicht mit einem Netzwerk verbunden sein.
- Es sind keine weiteren PCs erforderlich, um die Leistung zu erhöhen.
- Der vorhandene Prozessor kann relativ leicht hinsichtlich seiner Mehrkerneigenschaften (Multithreading: das parallele Abarbeiten mehrerer Threads) untersucht und die Zahl der parallel ausgeführten Simulationen angepasst werden.
- Es ist keine Lastverteilung erforderlich, da die Kerne in der Regel die gleiche Leistungsfähigkeit besitzen.
- Die Verteilung der parallelen Simulationen auf die vorhandenen Kerne wird vom Betriebssystem übernommen.
- Die aktuelle Entwicklung auf dem Prozessormarkt deutet darauf hin, dass künftig 4, 8 oder mehr Kerne in einem handelsüblichen Prozessor untergebracht werden.
- Selbst wenn nur ein Kern zur Verfügung steht, ist durch das parallele Ausführen der Simulationen ein nicht zu vernachlässigender Geschwindigkeitsvorteil zu verzeichnen, da dieser in Zusammenarbeit mit dem Betriebssystem und den Festplatten besser ausgenutzt werden kann.

Nachteilig wirken sich diese Punkte aus:

- Die Zahl der Kerne lässt sich nicht beliebig erhöhen lässt.
- Es werden höhere Anforderungen an die Programmierung gestellt.

-
- Es können schwere Speicherzugriffsverletzungen entstehen, falls der Code nicht klar strukturiert ist.

Nach Abwägung der Vor- und Nachteile wurde in dieser Arbeit die unter **Punkt 4** beschriebene Methode, das parallele Ausführen von zwei oder mehreren Simulationen auf einem Mehrkernprozessor realisiert und der bestehende Optimierungsalgorithmus um die erforderlichen Funktionalitäten erweitert.

5.2 Grundlagen zur Implementierung

Die Parallelisierung wird mithilfe der Threads vorgenommen. Sie werden auch als Ausführungsstränge betrachtet, der Teil eines Prozesses ist (Balena, 2004). Multithreading bezeichnet das gleichzeitige Abarbeiten mehrerer Threads. Dabei teilen sich mehrere Threads einen gemeinsamen Adressraum (Shared Memory) und werden auf denselben oder verschiedenen Prozessoren eines Multiprozessorsystems ausgeführt. Multithreading hat die Aufgabe Parallelismus in Multiprozessorsystemen mit modernen Prozessoren auszuführen (Rauber und Rüniger, 2000). Multithreading setzt aber nicht voraus, dass mehrere Prozessoren physikalisch vorhanden sind. Auch bei Einkernprozessoren werden Threads verwendet, um Anwendungen scheinbar parallel abzarbeiten.

In dem hier vorliegenden Fall werden die von .NET bereitgestellten Funktionalitäten verwendet, um Threads zu starten und zu steuern. In der Literatur wird mehrfach und ausdrücklich darauf hingewiesen, dass die Verwendung von Threads zu nicht nachvollziehbaren und gegebenenfalls schweren Systemabstürzen führen kann (MacDonald, 2006; Löffelmann, 2006). Der Grund hierfür liegt in der Shared Memory (dt. „gemeinsamer Speicher“) Architektur. Die auf dem Prozessor laufenden Threads teilen sich einen bestimmten Bereich des Arbeitsspeichers (RAM: Random Access Memory) eines PCs. Sie können auf den gleichen Adressraum mit normalen Speicherzugriffsoptionen zugreifen. Infolgedessen können schwere Zugriffsverletzungen entstehen, wenn beide Threads in den gleichen Speicherbereich schreiben möchten. Die Folge ist eine gegenseitige Blockade, welche zum Absturz der Anwendung führt. Um diese zu verhindern, gilt es bei der Parallelisierung den gemeinsamen erforderlichen Adressraum möglichst klein zu halten und die Threads in der Art zu synchronisieren, dass Zugriffe auf gemeinsame Speicherbereiche kontrolliert erfolgen.

Falls ein Objekt von mehr als einem Thread bearbeitet wird, kann es der Fall sein, dass zwei der Threads exakt gleichzeitig auf das Objekt zugreifen. Wir nehmen an, dass einen globaler Zähler von mehreren Threads verwendet wird. Der Zähler überwacht, wie oft eine Operation bereits ausgeführt wurde. Früher oder später möchte der Thread A den Wert von 10 auf 11 erhöhen und zur gleichen Zeit möchte Thread B den Wert von 10 auf 11 erhöhen. Das Ergebnis wird in dieser Situation 11 sein, obwohl der richtige Wert 12 wäre (Balena, 2004). Dies geschieht, da beide den Anfangswert 10 lesen, diesen um eins erhöhen und 11 in den Speicher schreiben. Je mehr Threads parallel ablaufen und je größer die Zeit zwischen dem Lesen und Schreiben eines Zählers ist, desto häufiger tritt dieses Problem auf (MacDonald, 2006).

Solche Synchronisationsprobleme sind häufig und problematisch, da sie nicht wie bei sequenziellen Programmabläufen auftreten müssen, sondern unter Umständen erst in der Anwendung und nicht beim Testen auftreten. Im Gegensatz zu sequenziellen Programmabläufen spielt nicht nur die Reihenfolge, sondern auch die Zeit eine Rolle. Ein Objekt (genauer einen Speicherbereich) von mehreren Threads nutzen zu lassen stellt zunächst kein Problem dar, wenn der Zugriff von beiden Kernen nicht exakt gleichzeitig erfolgt. Ist dies aber zufällig der Fall, ist die Konsequenz der Abbruch der Anwendung. Die Parallelisierung der Evaluierung erfordert eine gut strukturierte Programmierung, vor allem da auch das Debuggen erheblich mühsamer wird. Ein schrittweises Überprüfen des Algorithmus ist äußerst gewöhnungsbedürftig, da während der Fehleruntersuchung eines Threads, ein anderer im Hintergrund weiter arbeitet. Der Programmablauf ist zwar noch linear, aber es werden mehrere Stränge parallel verfolgt, so dass der Zeitpunkt der Ausführung der Methoden unbekannt ist.

5.3 Implementierung der Parallelisierung

Oben beschriebene Probleme wie beispielsweise das Shared-Memory-Problems und das Synchronisationsproblem wurden gelöst, indem unter anderem der parallele Anteil kleinstmöglich gehalten wurde. Die Vorbereitung der Simulationsmodelle, die Überprüfung der Zustände der Threads und die Verarbeitung der Simulationsergebnisse erfolgten gänzlich sequenziell. Lediglich die Simulation mit dem BluM.Sim Modell erfolgt parallel. Dieses Vorgehen widerspricht zwar der Forderung des Amdahlschen Gesetzes, einen möglichst großen Anteil zu parallelisieren, aber auf diese Weise konnten die Dead-Locks beim Zugriff

auf den Shared-Memory verhindert werden. Alle anderen Versuche die Variablen zu kapseln und größere Teile des Controllers zu parallelisieren erweisen sich als nicht praktikabel. Außerdem sind die erforderlichen Rechenzeiten für die sequenzielle Vorbereitung um Größenordnungen kleiner als die erforderliche Rechenzeit für die Simulation ($T_S \ll T_P$).

Das Starten der Threads erfolgt in einer „Do Loop While“ Schleife, welche so lange wiederholt wird, bis alle Nachfahren einer Generation evaluiert wurden. Hierzu war es erforderlich, die Prozeduren des nominell diskreten Algorithmus 3-1, des reellwertigen Algorithmus 2-1 und des memetischen Algorithmus 3-4 so zu gestalten, dass alle Nachfahrenindividuen zur Evaluierung einer gesamten Generation zur Verfügung stehen. Die Zahl der zu verwendenden Threads wird automatisch an die Zahl der vorhandenen Prozessoren und Prozessorkerne innerhalb eines Prozessors angepasst. Die Verwendung von mehreren If-Abfragen ermöglicht die Priorisierung der abzuarbeitenden Funktionen.

```
1  Do Until „Alle Nachfahren wurden evaluiert“
2      If „ein Thread ist frei“ Then
3          Berechne die Evaluierung vor
4          Evaluiere
5      Else If „ein Thread ist beendet und die Evaluierung war erfolgreich“ Then
6          Verarbeite das Ergebnis
7          Ermittle den Qualitätswert
8      Else If „ein Thread ist beendet und die Evaluierung war nicht erfolgreich“ Then
9          Kennzeichne diesen Nachfahren und evaluiere gesondert
10     Else
11         Warten für 400ms
12     End If
13 Loop
```

Algorithmus 5-1: Prozedur der Parallelisierung der Simulationen innerhalb jeder Generation

Der im obigen Algorithmus 5-1 dargestellte Pseudocode entspricht in allen Prozeduren der Zeile: *Evaluieren*: P_{ij}^{***} . Darin hat immer die Vorbereitung und das Starten der Threads Vorrang, um die vorhandenen Ressourcen des Rechners auszunutzen. Es wird also versucht, immer die maximale Anzahl von Threads auszuschöpfen, unabhängig davon, ob ein bestimmter Thread erfolgreich evaluiert bzw. vom Simulationsmodell fehlerfrei simuliert

wurde und unabhängig davon, ob gegebenenfalls das Ergebnis ausgewertet wurde. Mit der Priorität zwei erfolgt die Auswertung der Threads. Allerdings erfolgt die Auswertung nur dann, wenn das Simulationsmodell fehlerfrei simuliert hat. Mit Priorität drei werden fehlerhafte Simulationen berücksichtigt. Diese werden vermerkt und später gesondert abgearbeitet. Falls aktuell die ersten drei Fälle nicht zutreffend sind, fällt diese sequenzielle „Do Loop While“ Schleife für 400 Millisekunden in einen Ruhemodus. Dies ist erforderlich, da sonst der Prozessor mit dem fortwährenden Durchlaufen der Schleife ausgelastet wäre und die Threads nicht ausgeführt werden könnten.

Die vorher beschriebenen Synchronisationsprobleme wurden gelöst, in dem alle erforderlichen Variablen (im speziellen die „Zähler“) vom sequenziellen Programmteil gehalten und verwaltet werden. Die parallel arbeitenden Threads benötigen keinen Zugriff auf Variablen, die für die Prozedursteuerung nötig sind. Der sequenzielle Programmteil übernimmt auch die Vorbereitung der Modelldatensätze (Preprocessing) und das Verarbeiten der Modelldatensätze (Postprocessing) (siehe Abbildung 5-2). Die Threads erhalten lediglich die Information darüber, welcher Datensatz evaluiert und wann mit der Simulation begonnen werden soll. Durch Anfragen an die Simulationsverwaltung (BlueM.Sim Control) erhält der sequenziell arbeitende Optimierungsalgorithmus die erforderlichen Informationen bezüglich der Status der einzelnen Threads. Dafür mussten alle Prozeduren der Algorithmen erweitert werden. Insbesondere wurde die Prozedur des reellwertigen Algorithmus 2-1 so verändert, dass wie bei den anderen Algorithmen immer eine ganze Generation P_i von Individuen evolutioniert wird um diese parallel evaluieren zu können.

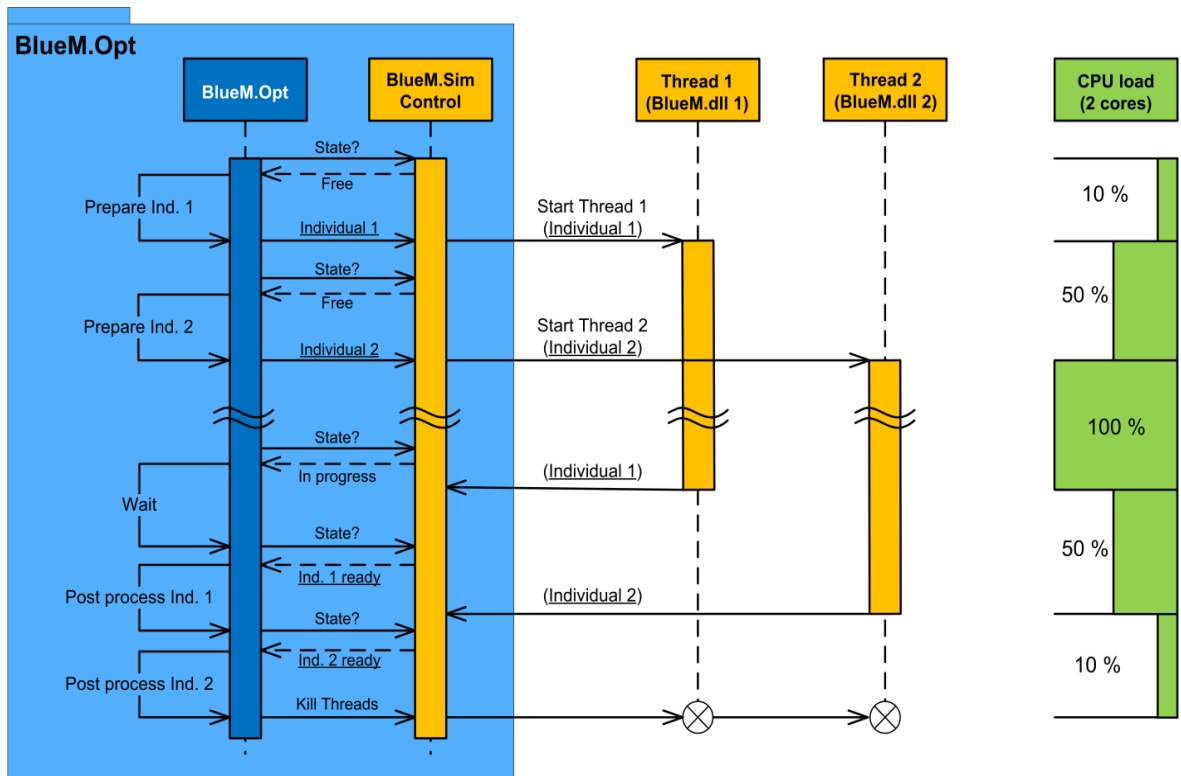


Abbildung 5-2: Sequenzdiagramm zweier Threads

Mit der oben beschriebenen Schleife und den Funktionen *ThreadFree* und *ThreadReady* fragt der Optimierungsalgorithmus *BlueM.Opt* fortlaufend den Controller *BlueM.Sim* nach dem Status der Threads.

Die Funktion *ThreadFree* fragt den Controller, ob ein Thread verfügbar ist und gestartet werden kann. Falls *JA* wird das entsprechende Thread-Verzeichnis vorbereitet und das zu evaluierende Individuum (der Nachfahre) an den Thread übergeben (Preprocessing). Mit *launchSim* (Start Thread 1) wird der Thread (die Simulation) gestartet. Sofort danach wird der zweite Thread gestartet.

Die Funktion *ThreadReady* fragt die Threadverwaltung, ob ein Thread beendet ist. Falls *JA* werden die Ergebnisse aus dem entsprechenden Verzeichnis verarbeitet (Postprocessing). Des Weiteren werden hier zwei Fälle unterschieden. Je nachdem ob die Simulation fehlerfrei erfolgte (*SimOK*) wird die Simulation entweder verarbeitet oder später wiederholt.

LaunchSim (überladen) führt die Simulation aus und kann entweder für die sequenzielle Simulation (ohne Parameter) oder für die parallele Simulation (mit Parameter) für die Threads verwendet werden.

5.4 Realisierter Geschwindigkeitsgewinn

Die Parallelisierung wurde mit dem Modelldatensatz der Erft auf einem Prozessor mit zwei physikalischen Kernen erprobt. Bei dem ermittelten sequenziellen Anteil von 1,8 % kann, dem Amdahlschen Gesetz (5-4) zufolge ein maximaler Geschwindigkeitsgewinn von 1,96 erwartet werden. Abbildung 5-3 zeigt die gemessenen Ergebnisse bei der Verwendung von 1 – 4 Threads.

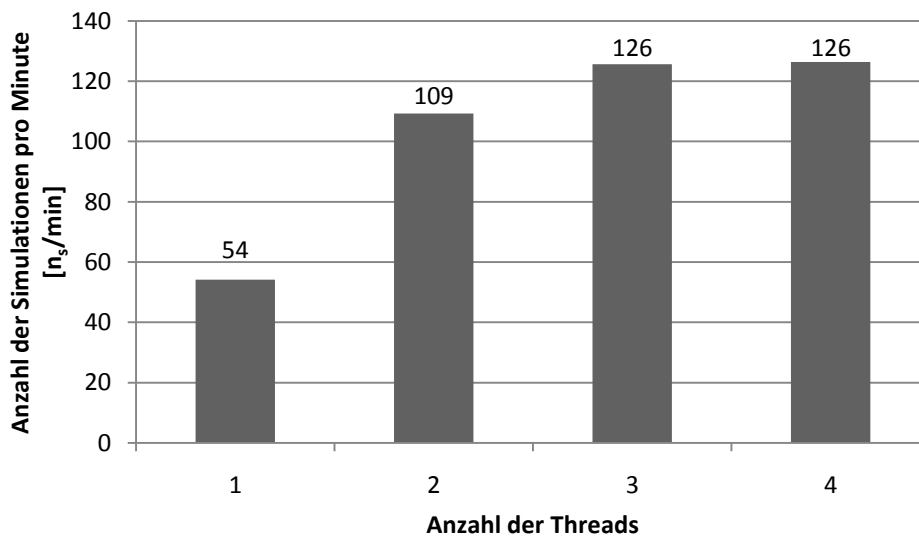


Abbildung 5-3: Anzahl der Simulationen pro Minute durch Parallelisierung der Simulationen bei Verwendung einer 2-Kern-CPU

Abbildung 5-3 zeigt, dass mit einem Thread 54 Simulationen pro Minute ausgeführt werden können. Werden zwei Threads verwendet, werden wie erwartet fast doppelt so viele Simulationen ausgeführt. Interessant ist allerdings, dass bei drei und mehr parallel ausgeführten Simulationen der Faktor nicht 2 ist, sondern 2,3. Der Grund hierfür ist das sogenannte *Simultaneous Multithreading* (SMT). Simultaneous Multithreading bezeichnet die Fähigkeit eines Mikroprozessors, mittels getrennter Abarbeitung der Maschinenbefehle und/oder zusätzlicher Registersätze mehrere Threads gleichzeitig auszuführen. Ein Thread bzw. eine Modellsimulation benötigt nicht immer alle Einheiten (numerische Coprozessoren, Vektorarithmetikeinheiten, Cache-Speicher, etc.). Falls mehrere Threads vorhanden sind, erlaubt Simultaneous Multithreading die ungenutzten Einheiten parallel anzusprechen. Außerdem wird deutlich, dass eine höhere Anzahl von Threads als vorhandene physikalische Kerne sich nicht negativ auf die Optimierungsgeschwindigkeit auswirken.

Die Ergebnisse zeigen das Potenzial der Parallelisierung. Dem Amdahlschen Gesetz zufolge (Gleichung (5-5)) wäre in diesem Beispiel für eine theoretisch unbegrenzte Anzahl vorhandener Prozessoren eine Beschleunigung der Optimierung um den Faktor 55 möglich. Die Ergebnisse zeigen aber auch, dass das Amdahlsche Gesetz an Gültigkeit verloren hat, da bedingt durch ihre Architektur (Simultaneous Multithreading) auch Mikroprozessoren mit einem Kern die Optimierung beschleunigen, falls diese parallel ausgeführt werden.

6 Anwendungsbeispiel Erft

Die oben beschriebenen Algorithmen werden in dem folgenden Anwendungsbeispiel an der Erft erprobt. Nach einer kurzen Beschreibung des Einzugsgebietes werden die Pegel aufgelistet, die als Eingangsgrößen zur Berechnung des Wellenablaufs im Gewässer verwendet werden. Im Einzugsgebiet existieren derzeit vier Hochwasserrückhaltebecken. Ein weiteres Rückhaltebecken ist in Planung. Neben den bestehenden Standorten werden 8 weitere Standorte für Hochwasserrückhaltebecken im Haupt- und im Nebenschluss untersucht.

Die Zielsetzung des Erftverbandes ist es im Einzugsgebiet einen 100-jährlichen Hochwasserschutz herzustellen. Die Zielsetzung der bereits realisierten Maßnahmen ist der örtliche Hochwasserschutz. Bei dem Hochwasserereignis im Sommer 1984 kam es durch die Verzögerung der Abflüsse an der Erft, bedingt durch die Abgaben nach dem Einstau des Hochwasserrückhaltebeckens Eicherscheid, beim Zusammenfluss mit dem Swistbach zu einer Erhöhung des Abflussscheitels. Im Nachgang der Studie von Huebner (2003) wurde der Betriebsplan des Hochwasserrückhaltebeckens modifiziert, um diese „ungünstige“ Steuerung zu vermeiden. Die Problematik bleibt allerdings bestehen, da die Hochwasserwelle des Swistbaches tendenziell später an der Mündung in die Erft eintrifft als die Hochwasserwelle der Erft.

6.1 Das Einzugsgebiet der Erft

Die Erft mit einer Einzugsgebietsfläche von 1.828 km² entspringt auf einer Höhe von 520 m+NN südlich von Bad Münstereifel in der Osteifel und mündet nach einer Fließstrecke von 107 km im Niederrheinischen Tiefland bei Neuss auf ca. 26 m+NN in den Rhein (Abbildung 6-1). Das Einzugsgebiet erstreckt sich über zwei geologische Gebiete: dies sind die Eifel mit anstehenden Festgesteinen und die Niederrheinische Bucht mit Lockersedimenten.

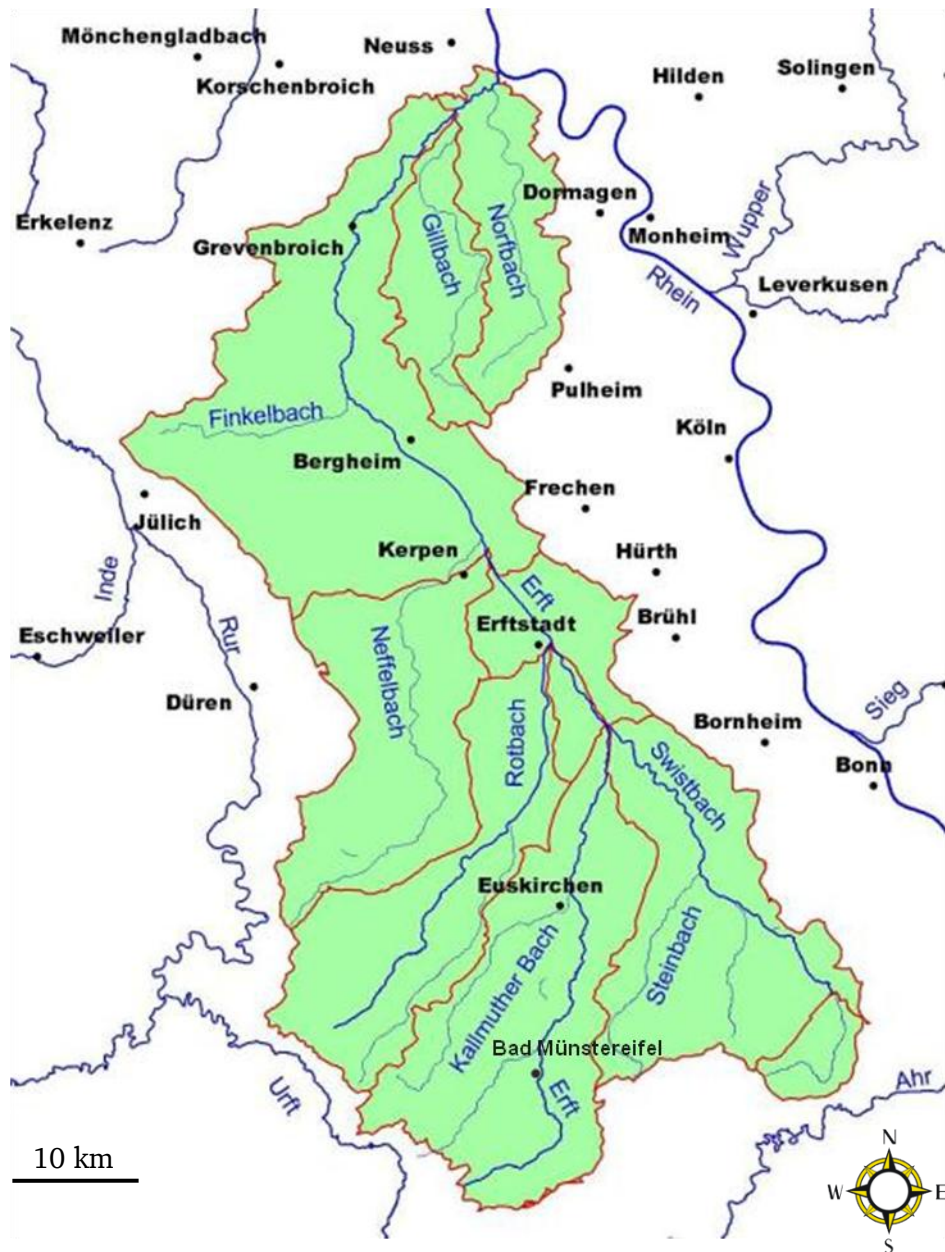


Abbildung 6-1: Einzugsgebiet der Erft (Erfstverband, 2008)

Der mittlere Jahresniederschlag beträgt ca. 670 mm. Die Hauptniederschlagszeit ist der Sommer, die Wintermonate sind niederschlagsärmer. Die Erft teilt sich im Mittellauf in die Kleine Erft, Große Erft und den Erftkanal. Die Kleine Erft zweigt bei Brüggeln nach Westen ab, kreuzt auf der Höhe von Kerpen den Erftkanal mit einem Aquädukt und fließt nach 19 km in Bergheim zurück in die Erft. Die Große Erft zweigt zwischen Sindorf und Horrem ebenfalls nach Westen ab und mündet nach 7,3 km wieder in der Erft. Die Swist, der

Gillbach und die Norf fließen von Südosten zu, der Veybach, der Rotbach und der Neffelbach von Südwesten.

Tabelle 6-1: Gewässersteckbrief (Ministerium fuer Umwelt und Naturschutz, Landwirtschaft und Verbraucherschutz NRW, 2009)

Oberirdische Einzugs- gebietsgröße	1.828 km ²
Lauflänge	106,6 km
Höhenlage	520 m+NN bis 26 m+NN
Mittleres Gefälle	0,46 %
Mittlere Jahresnieder- schlagshöhe	670 mm
Flussgebietseinheit	Rhein
Zuflüsse im EZG Größe > 10 km²	Norfbach, Gillbach, Elsbach, Pützbach, Finkelbach, Wiebach, Fisch- bachgraben, Neffelbach, Rotbach, Swistbach, Lommersumer Mühlengraben, Kuchenheimer Mühlengraben, Veybach, Mersbach, Eschweiler Bach
Geologie	Südlicher Teil (20 %): Eifel mit anstehenden Festgesteinen (Schiefer, Sandstein, Kalke), nördlicher Teil (80 %): Nieder- rheinische Bucht mit marinen und terrestrisch-fluviatilen Sedi- menten, teilweise mit bis zu 100 m mächtigen Braunkohlenflözen in Wechsellagerung mit Sanden, Kiesen und Tonen
Strömungsenergie	Die Abflussverhältnisse sind durch Querbauwerke und Ausbaumaß- nahmen verändert. Im Bereich Sumpfungswassereinleitungen (siehe folgendes Kapitel Anlage D „Erftverlegungen“) sind die natürlichen Abflussschwankungen stark gedämpft. Die Abfluss- mengen sind für den Zeitraum der Sumpfungmaßnahmen deutlich vergrößert.
Durchschnittliche Wasser- breite (Ausbauzustand)	Oberlauf: 3 m; Mittellauf: 6 m; Unterlauf: 17–25 m
Durchschnittliche Wasser- tiefe	Oberlauf: 0,15 m; Mittellauf: 0,6 m; Unterlauf: 1,25–2,5 m
Talform	Quelle bis oberhalb Euskirchen: Auen- und Muldentalgewässer Euskirchen bis Erftstadt: Kiesgewässer Erftstadt bis Mündung: Tieflandgewässer
Flächennutzung	Wald: 16 %, Grünland: 4 %, Städtische Flächen: 14 %, Landwirt- schaft: 67 %

Das Pegelmessnetz umfasst 26 Pegel, die teilweise vom Land, teilweise vom Erftverband betrieben werden. Der überwiegende Teil der Landespegel und ein Teil der Pegel des Erftverbandes sind mit Datenfernübertragungsmöglichkeiten ausgerüstet. Hinsichtlich der Datenverfügbarkeit ist das wasserwirtschaftliche System gut ausgebaut.

6.2 Pegel und Hochwasserereignisse

Aus den Pegelmessungen der Jahre 1969 bis 2002 wurden drei Hochwasserereignisse ausgewählt. Voraussetzung war, dass vollständige Zeitreihen für die Ereignisse an folgenden Pegeln (Abbildung 6-2) vorlagen: Friesheim, Hausweiler, Langenich, Moedrath, Schoenau, Veynau, Weilerswist.

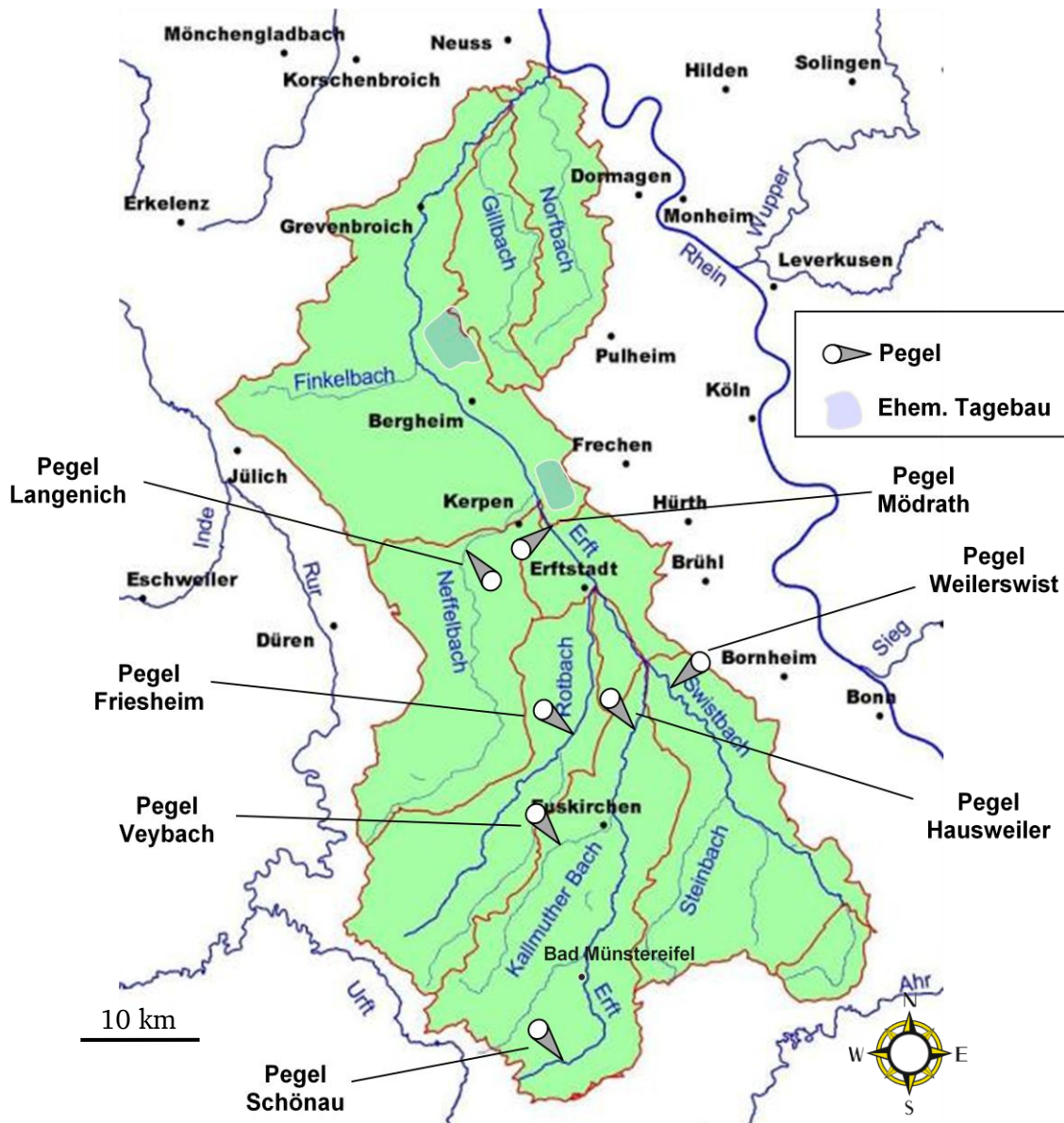


Abbildung 6-2: Lage der als Eingangsinformation verwendeten Pegel

Das erste Ereignis ist aus dem Sommer 1984, hat zwei Scheitel und ist das größte betrachtete Ereignis (Abbildung 6-3).

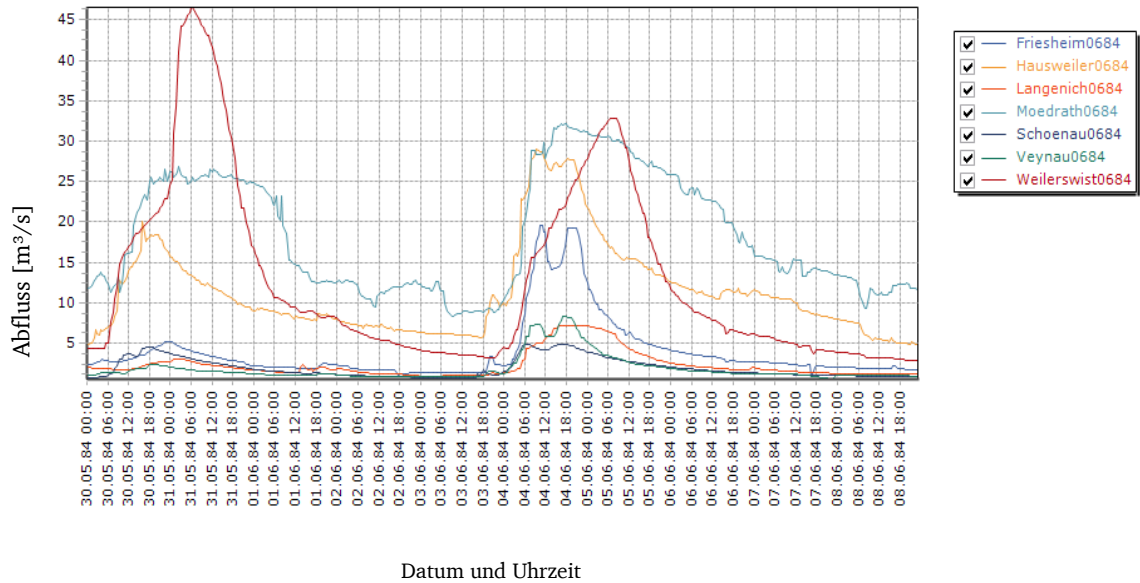


Abbildung 6-3: Abflüsse des Ereignisses Juni 1984 an den 7 Pegeln

Das zweite Ereignis ist ein mehrgipfliges Winterereignis aus dem Jahr 1995 (Abbildung 6-4).

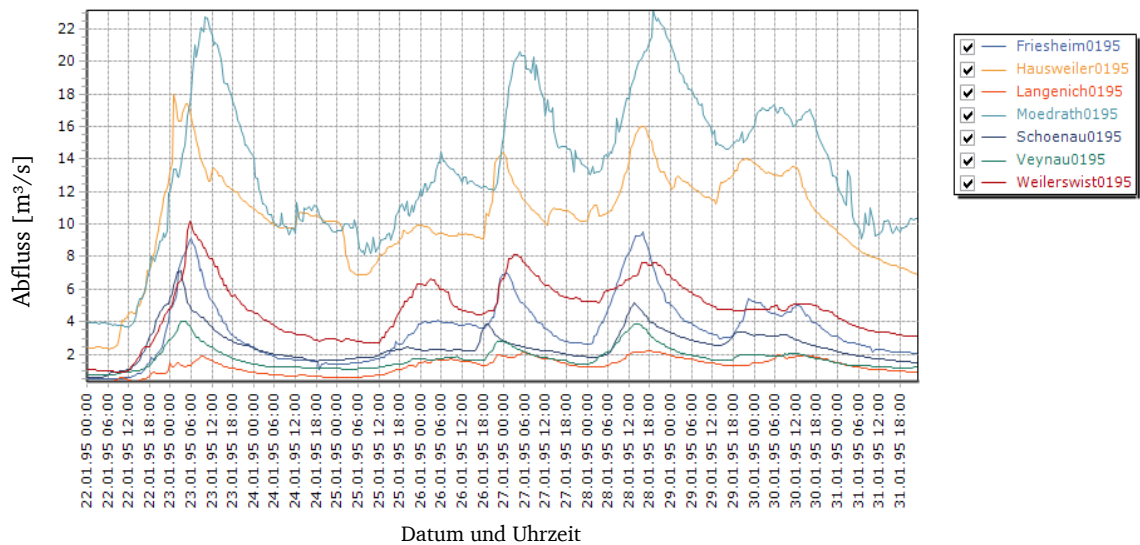


Abbildung 6-4: Abflüsse des Ereignisses Januar 1995 an den 7 Pegeln

Das dritte Ereignis ist ein stammt aus dem Jahr 2000 und hat einen ausgeprägten Hochwasserscheitel (Abbildung 6-5).

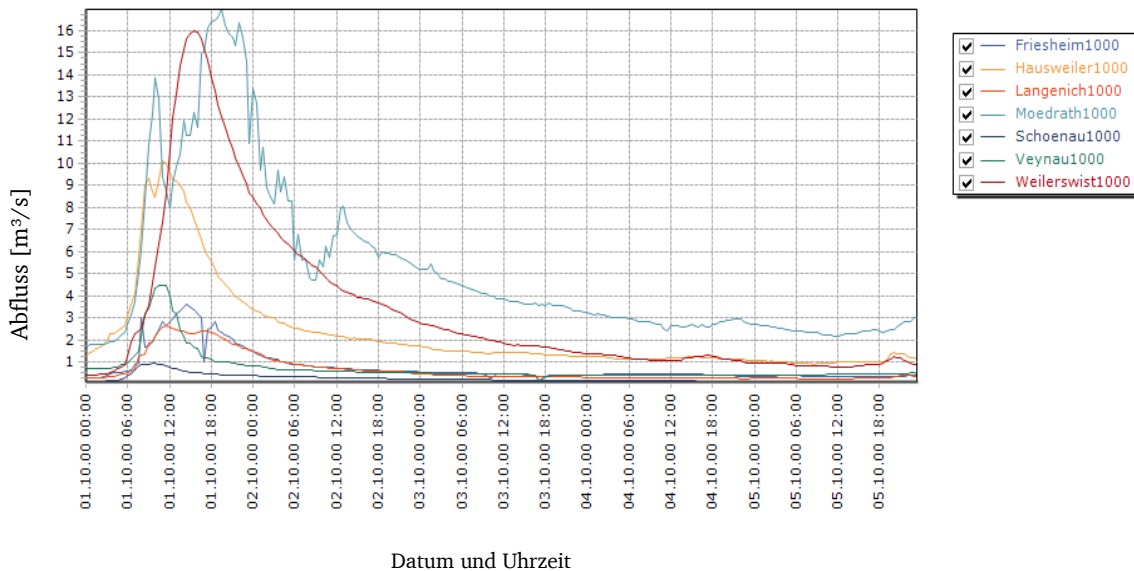


Abbildung 6-5: Abflüsse des Ereignisses Oktober 2000 an den 7 Pegeln

Die Abflüsse ausgewählter Scheitel sind in folgender Tabelle 6-2 aufgetragen:

Tabelle 6-2: Betrachtete Hochwasserwellen

Hochwasser	Gemessen am Pegel	Datum [d]	Uhrzeit [h]	Abfluss Q_{\max} [m ³ /s]	Wiederkehrintervall [a] (Schütz, 2007)
1984/06	Weilerswist	31.05.1984	06:14	46,5	70
1984/06	Weilerswist	05.06.1984	07:45	32,3	
1995/01	Mödrath	23.01.1995	10:00	22,7	5
1995/01	Mödrath	27.01.1995	4:30	20,5	
1995/01	Mödrath	28.01.1995	19:00	23,0	
2000/10	Weilerswist	01.10.2000	19:30	17,0	3

Da das Ereignis aus dem Jahr 1984 den größten Abflussscheitel aufweist wird es für die meisten Betrachtungen verwendet. Die beiden anderen Ereignisse werden bei den Optimierungsergebnissen herangezogen, um zu untersuchen, ob es möglich ist den Hochwasserschutz bei Ereignissen mit unterschiedlichen Wiederkehrintervallen zu erhöhen.

6.3 Bestehende Hochwasserschutzmaßnahmen

Derzeit sind im Einzugsgebiet der Erft die Hochwasserrückhaltebecken Eicherscheid, Horchheim, Niederberg und Mödrath in Betrieb (Erftverband, 2006; 2007). Für das Rückhaltebecken Garsdorf liegt die Genehmigung seit November 2000 vor. Die Realisierung ist abhängig vom Zeitpunkt der vollständigen Rekultivierung des Tagebaus Fortuna-Garsdorf (Erftverband, 2009). Die Rückhaltebecken Horchheim und Eicherscheid befinden sich im

Oberlauf der Erft (siehe Abbildung 6-6) und besitzen zusammen ein Retentionsvolumen von 2.477 Mio. m³. Das Rückhaltebecken Niederberg, mit 0,9 Mio. m³ Stauraum liegt im Mittellauf des Rothbachs, das Becken Mödrath mit 1,7 Mio. m³ Stauraum im Mittellauf der Erft. Das planfestgestellte Rückhaltebecken Garsdorf ist ebenso wie Mödrath ein ehemaliger Braunkohletagebau mit einem Volumen von 1.6 Mio. m³ und liegt im Unterlauf.

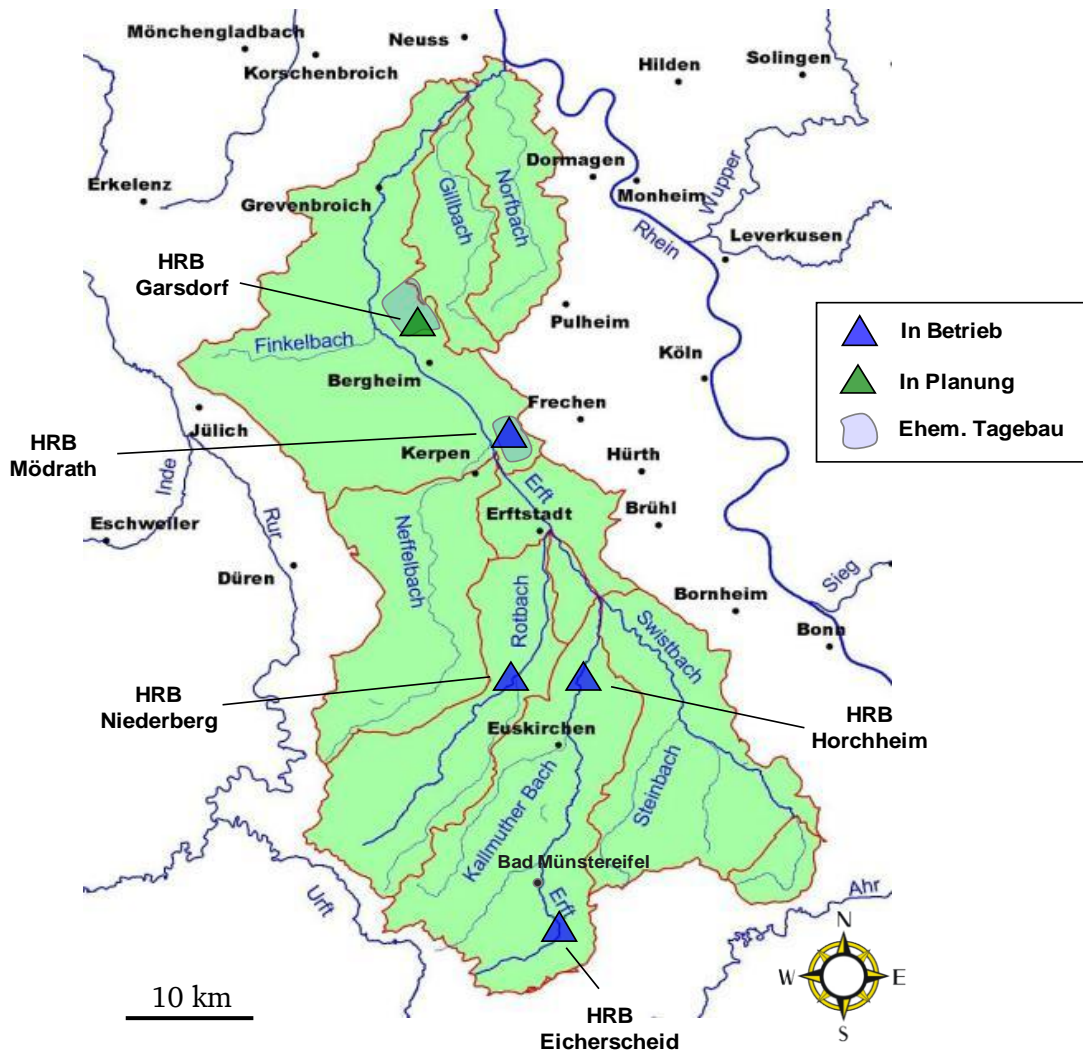


Abbildung 6-6: Lage der geplanten und in Betrieb genommenen Rückhalteräume

Die Becken Garsdorf und Mödrath liegen im Nebenschluss, Horchheim und Eischerscheid im Hauptschluss an der Erft. Das Becken Niederberg liegt im Hauptschluss am Rotbach.

Für die Hochwasserrückhaltebecken Horchheim, Eischerscheid, Niederberg und Mödrath lagen Betriebsregeln vor. Diese wurden bei der Modellierung berücksichtigt. Für das Becken Garsdorf liegt kein Betriebsplan vor. Da aber das Volumen bereits bekannt ist, wurde das Becken berücksichtigt und mit einer einfachen Betriebsregel versehen.

Tabelle 6-3: Bestehende und geplante Hochwasserrückhaltebecken (Erftverband, 2006; 2007; 2008; 2009)

HRB	Eicherscheid	Horchheim	Niederberg	Mödrath	Garsdorf
A_E [km ²]	38	281	175	944	/
V_g [Mio. m ³] (ohne V_{Dauer})	1,012	1,465	0,920 (erf. Vol.)	1,700	1,600
V_{Dauer} [Tsd. m ³]	30	0	0	0	/
HQ_{100} ([m ³ /s]	43,00	65,00 (Wert ohne HRB Eicherscheid. Mit HRB E.: 58,3)	37,00	44	/
HQ_{1000} [m ³ /s]	70,00	90,00	60,00	/	/
Max. Q_{ab} [m ³ /s]	7,0	30,0	20,0 (konstant)	0 - 3,0 (stufenlos regelbar)	0 - 5,5 (stufenlos regelbar)
MQ [m ³ /s]	0,42	1,62	0,6	/	/
Gewässer	Erft	Erft (und Mühlengraben)	Rotbach	Erft	Erft
Sohlenhöhe [m+NN]	327,50	120,54	124,30	/	/
Investitionskosten [Mio. €]	/	/	7,8	7,3	7,2
Zustand	In Betrieb seit 1976	In Betrieb seit 1984	In Betrieb seit 2006	In Betrieb seit 2007	Plan festgestellt

6.4 Zusätzliche Maßnahmen zur Optimierung

Eine Voraussetzung ist, dass potenzielle Lagen für eine oder mehrere Maßnahmen bekannt sind. Daher müssen die Standorte vorab festgelegt werden, um zu untersuchen, ob ein Standort dem Hochwasserschutz zuträglich ist oder nicht. Soll verglichen werden, ob ein Standort A oder B effizienter ist, um den Hochwasserscheitel zu reduzieren, müssen an beiden Positionen Maßnahmen modelliert werden.

Die Standortauswahl in dieser Arbeit erfolgte auf der Basis von topografischen Karten und GIS-Daten. Dabei wurden auch aktuelle Planungen des Erftverbandes berücksichtigt, wie beispielsweise das Hochwasserrückhaltebecken Garsdorf. Als weiteres nützliches Werkzeug zur Standortsuche erwies sich die Software GoogleEarth™. Google Earth ermöglicht die Überlagerung von digitalen Geländemodellen mit Satellitenfotos, so dass die Umgebung 3-dimensional aus der Vogelperspektive betrachtet werden kann.

Grundsätzlich ist es mit dem Memetischen Algorithmus möglich, Hochwasserspeichersysteme aller Art zu optimieren, sofern diese Maßnahmen mit dem verwendeten Simulationsmodell abgebildet werden können. Eine Übersicht über Hochwasserschutzmaßnahmen im Allgemeinen von Patt und Bechteler (2001) befindet sich in der Anlage E dieser Arbeit. Ein Verfahren zur Optimierung des Gerinnes (Breite und Lage der Sohle) unter Berücksichtigung der Kosten wurde von Feldmann (2009) entwickelt.

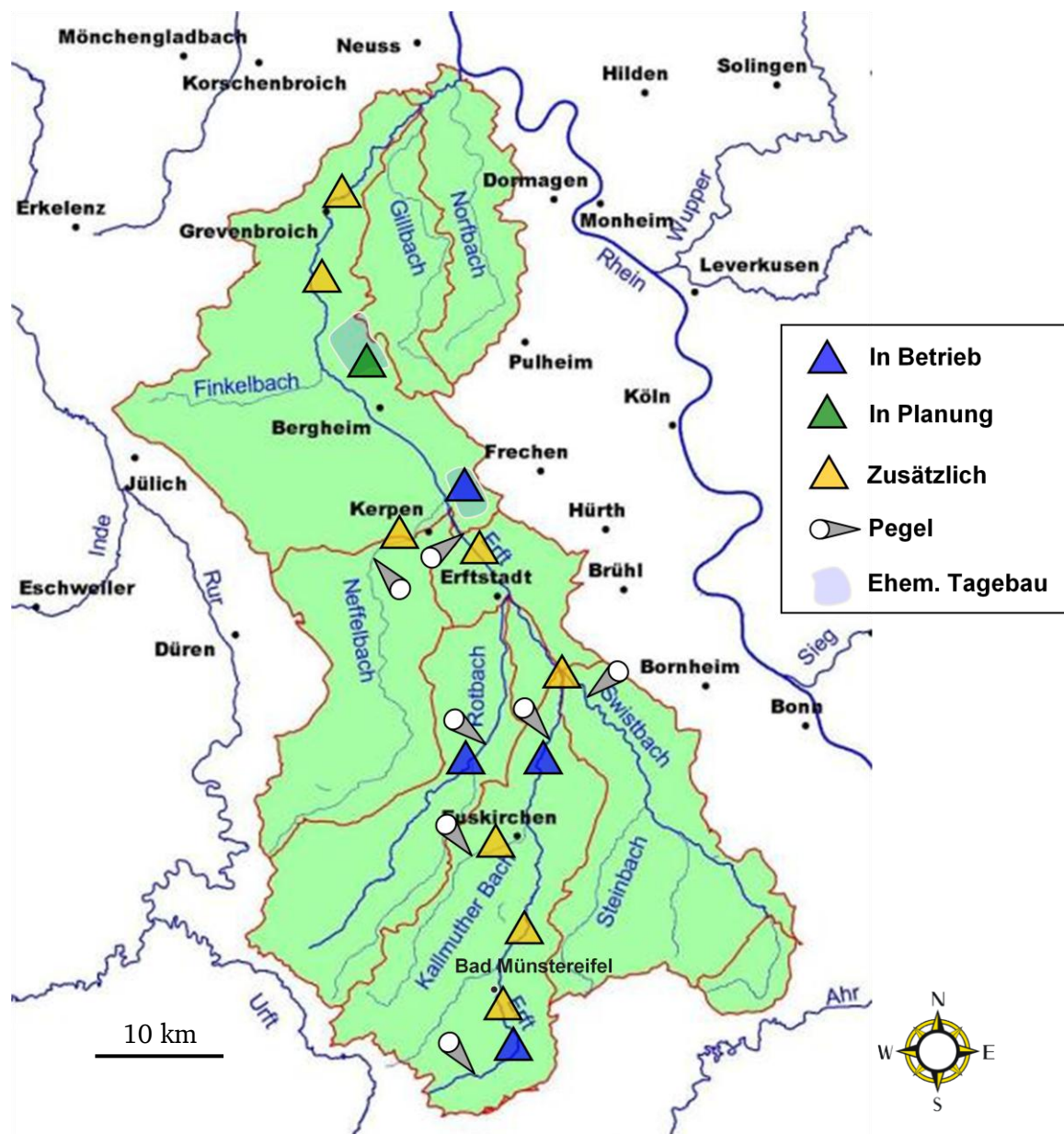


Abbildung 6-7: In der Optimierung zusätzlich betrachtete Standorte für Hochwasserspeicher

Im Einzugsgebiet der Erft werden im wesentlichen Hochwasserrückhaltebecken erbaut um künftig einen 100 jährlichen Hochwasserschutz herzustellen. Bei der Optimierung wurden

zusätzlich 13 weitere Standorte für Hochwasserrückhaltebecken berücksichtigt (Abbildung 6-7).

Die folgende Abbildung 6-8 zeigt den Systemplan der zur Modellierung und Optimierung der Maßnahmen erstellt wurde. Die bestehenden Hochwasserrückhaltebecken sind mit A-D gekennzeichnet. Die zusätzlichen Standorte mit S1-S9. In den mit *M* gekennzeichneten Bereichen werden die Sumpfungsmaßnahmen für den Braunkohletagebau berücksichtigt (siehe dazu auch Anlage D „Erftverlegungen“). Die Verzweigungen dienen als Umschalter zwischen mehreren Maßnahmen an einem Standort oder verteilen den Abfluss bei Hochwasserrückhaltebecken im Nebenschluss zwischen Gerinne und Becken auf.

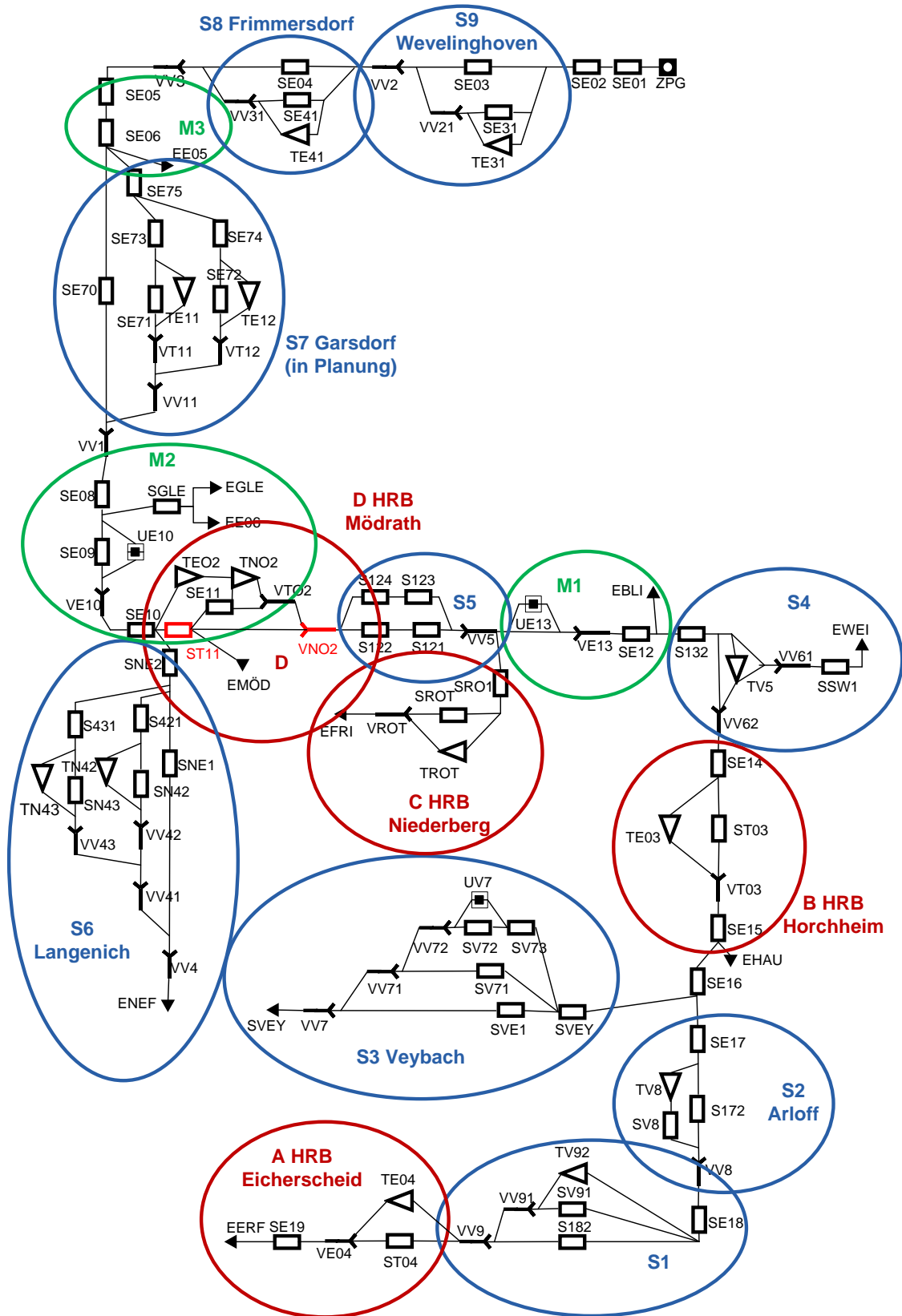


Abbildung 6-8: Optimierungssystemplan der Erft mit 9 Standorten (S1 bis S9)

Mit folgender Tabelle 6-4 werden die nominell diskreten Teile der Repräsentation durch die Verzweigungen im Modell abgebildet. In der ersten Spalte steht die Position, die zweite Spalte definiert die zugehörigen Maßnahmen. Die übrigen Spalten definieren wie die Verzweigungen eingestellt werden müssen um den entsprechenden Speicher zu aktivieren.

Tabelle 6-4: Auszug aus der Steuerungsdatei zur Kontrolle der nominelle diskreten Anteile

```

*Kombinatorik
*=====
*
* |-----|-----|-----|
* |           |           | Schaltung (Achtung Leerzeichen) |
* |-----|-----|-----|
* | Position   | Speicher   | VerZw 1 | VerZw 2 | VerZw 3 |
* |<----->|<----->|<--->|-x-|<--->|-x-|<--->|-x-|
| 0 HRB Eicherscheid | 0 Nullvariante ST04 | VE04 | 1 | X | X | X | X |
| 0 HRB Eicherscheid | 1 HRB Eicherscheid | VE04 | 0 | X | X | X | X |
| 1 Eicherscheid 2   | 0 Nullvariante S182 | VV9  | 1 | X | X | X | X |
| 1 Eicherscheid 2   | 1 Gerinne Mod       | VV9  | 0 | VV91 | 1 | X | X |
| 1 Eicherscheid 2   | 2 HRB Eicherscheid | VV9  | 0 | VV91 | 0 | X | X |
| 2 Arloff           | 0 Nullvariante S172 | VV8  | 1 | X | X | X | X |
| 2 Arloff           | 1 HRB Arloff        | VV8  | 0 | X | X | X | X |
| 3 Veybach          | 0 Nullvariante SVE1 | VV7  | 1 | X | X | X | X |
| 3 Veybach          | 1 Gerinne           | VV7  | 0 | VV71 | 1 | X | X |
| 3 Veybach          | 2 Gerinne, Wiese    | VV7  | 0 | VV71 | 0 | X | X |
| 35 HRB Horchheim   | 0 Nullvariante ST03 | VT03 | 1 | X | X | X | X |
| 35 HRB Horchheim   | 1 HRB Horchheim     | VT03 | 0 | X | X | X | X |
| 4 Weilerswist      | 0 Nullvariante S132 | VV61 | 1 | VV62 | 1 | X | X |
| 4 Weilerswist      | 1 HRB                | VV61 | 0 | VV62 | 0 | X | X |
| 45 HRB Niederberg  | 0 Nullvariante SROT | VROT | 1 | X | X | X | X |
| 45 HRB Niederberg  | 1 HRB Niederberg    | VROT | 0 | X | X | X | X |
| 47 Moedrath HRB    | 0 Nullvariante ST11 | VN02 | 1 | X | X | X | X |
| 47 Moedrath HRB    | 1 HRB Moedrath      | VN02 | 0 | X | X | X | X |
| 5 Moedrath Gerinne | 0 Nullvariante S121 | VV5  | 1 | X | X | X | X |
| 5 Moedrath Gerinne | 1 Gerinne           | VV5  | 0 | X | X | X | X |
| 6 Langenich        | 0 Nullvariante SNE1 | VV4  | 1 | X | X | X | X |
| 6 Langenich        | 1 PoldLangenich 0,62 | VV4  | 0 | VV41 | 1 | X | X |
| 6 Langenich        | 2 PoldLangenich 1,56 | VV4  | 0 | VV41 | 0 | X | X |
| 7 Garsdorf         | 0 Nullvariante SE70 | VV1  | 1 | X | X | X | X |
| 7 Garsdorf         | 1 HRB Garsdorf 900  | VV1  | 0 | VV11 | 1 | X | X |
| 7 Garsdorf         | 2 HRB Garsdorf 1600 | VV1  | 0 | VV11 | 0 | X | X |
| 8 Frimmersdorf    | 0 Nullvariante SE04 | VV3  | 1 | X | X | X | X |
| 8 Frimmersdorf    | 1 Füllung Restsee   | VV3  | 0 | X | X | X | X |
| 9 Wevelinghoven   | 0 Nullvariante SE03 | VV2  | 1 | X | X | X | X |
| 9 Wevelinghoven   | 1 Polder             | VV2  | 0 | X | X | X | X |
* |-----|-----|-----|

```

Folgende Tabelle 6-5 zeigt die Systemelemente aller Hochwasserspeicher und deren zu optimierende Parameter. Bei der Tabelle handelt es sich um einen Auszug aus der Steuerungsdatei um die reellen Parameter zu definieren.

Tabelle 6-5: Übersicht über die Speicher und deren Abgaben (Min,Max) und Volumina (Min,Max)

*Optimierungsparameter

*=====

*

* -----	-----	-----	-----	-----
* Bezeichnung	Einh.	Anfangsw.	Min	Max
* -----	-----	-----	-----	-----
TE04 Volumen	-	1010	202	2020
TV92 Volumen	-	280	56	560
TV08 Volumen	-	400	80	800
TE03 Volumen	-	1376	275	2752
TV05 Volumen	-	796	159	1592
TROT Volumen	-	1160	232	2320
TE02 Volumen	-	1700	340	3400
TN42 Volumen	-	612	122	1224
TN43 Volumen	-	1560	312	3120
TE11 Volumen	-	900	180	1800
TE12 Volumen	-	1600	320	3200
TE41 Volumen	-	1200	240	2400
TE31 Volumen	-	1000	200	2000
TE04 Abgabe	-	2	1	20
TV92 Abgabe	-	10	5	30
TV08 Abgabe	-	10	6	35
TE03 Abgabe	-	30	10	50
TV05 Abgabe	-	10	5	30
TROT Abgabe	-	20	5	40
TE02 Abgabe	-	3	1	15
TN42 Abgabe	-	1	1	10
TN43 Abgabe	-	1	1	10
TE11 Abgabe	-	1	1	10
TE12 Abgabe	-	1	1	10
TE41 Abgabe	-	0.8	0.5	10
TE31 Abgabe	-	0.8	0.5	10
* -----	-----	-----	-----	-----

Um die Optimierungsparameter in den Modelldatensatz zu übertragen ist eine weitere Steuerungsdatei erforderlich. Diese definiert in welche Dateien die Volumina und Abgaben geschrieben werden müssen. Ein Beispiel dazu ist in der Anlage G - 4 zu finden.

6.5 Zielsetzung der Optimierung

Die in der Einleitung beschriebenen Herausforderungen im Einzugsgebiet zeigen, dass es erforderlich ist, vorhandene und künftige Hochwasserspeicher nicht als singuläre Einheit mit örtlicher Wirkung zu betrachten, sondern stets das Gesamtsystem und damit die Interaktionen der Maßnahmen auf das Abflussgeschehen zu berücksichtigen. Die Abhängigkeiten der Maßnahmen sind komplex und können nicht mit einzelnen Modellrechnungen untersucht werden. Zu diesem Zweck sind definierte Zielsetzungen und Optimierungsalgorithmen erforderlich, um zu untersuchen, wie sich die Speicher im Verbund bei verschiedenen Belastungen verhalten.

Detaillierte Informationen zu Problemlagen in Bezug auf den Hochwasserschutz liegen für das Gewässer Erft nicht vor. Die Integration von maximal zulässigen Abflüssen, Wasser-

ständen oder Schadensfunktionen in das Optimierungssystem ist problemlos möglich. Beobachtungspunkte für den Abfluss sind im System beliebig platzierbar und mittels der Zielfunktionen können zulässige Abflüsse durch Referenzwerte berücksichtigt werden. Da diese Informationen nicht vorliegen, werden in den folgenden Optimierungsrechnungen an einem oder mehreren Positionen am Gewässer die maximalen Abflüsse betrachtet. Ziel dabei ist es den oder die Hochwasserscheitel soweit wie möglich zu reduzieren. Der Referenzwert wurde daher für alle Positionen zu null gesetzt.

7 Optimierungsergebnisse

Die Optimierungsergebnisse sind gegliedert in rein nominell diskrete Optimierungsläufe unter Verwendung des Algorithmus aus Kapitel 3.1 „Methodik der nominell diskreten Hochwasserspeicheroptimierung“ und des Algorithmus aus Kapitel 3.2 „Methodik der Memetischen Hochwasserspeicheroptimierung“. Die beiden Kapitel 7.1 und Kapitel 7.2 sind weiter untergliedert in monokriterielle und multikriterielle Problemstellungen.

Monokriterielle Problemstellungen berücksichtigen meist den Hochwasserscheitel an einem oder mehreren Zielpegeln. Wurden mehrere Zielpegel berücksichtigt, so wird die Reduzierung auf ein Ziel durch Aggregation erreicht. Bei multikriteriellen Problemen werden zusätzlich das Retentionsvolumen oder die Investitionskosten für die Maßnahmen berücksichtigt. In den meisten Optimierungsläufen wird das Ereignis aus dem Jahre 1984 (siehe Kapitel 6.2, Abbildung 6-3) simuliert um die Ergebnisse vergleichbar zu machen.

Um den Vergleich mit dem an der Erft aktuell bestehenden Hochwasserschutz zu ermöglichen wurde eine Referenzkombination von Maßnahmen bestehend aus den Becken Eicherscheid, Horchheim, Niederberg, Mödrath dem geplanten Becken Garsdorf berücksichtigt. Die Kombination ohne das Becken Garsdorf wird als Referenzszenario A und die Kombination mit dem Becken Garsdorf als Referenzszenario B bezeichnet.

7.1 Optimierungsergebnisse des nominell diskreten Algorithmus

Im folgenden Abschnitt werden die Optimierungsergebnisse des nominell diskreten Algorithmus bei Anwendung auf den Modelldatensatz für die Erft vorgestellt. Die Zielsetzung der monokriteriellen Optimierungsläufe ist die Reduzierung des Maximalabflusses Q [m³/s] am Zielpegel beziehungsweise an mehreren Pegeln im Einzugsgebiet. Die multikriteriellen Optimierungsläufe berücksichtigen zusätzlich das Retentionsvolumen und die Investitionskosten zur Erstellung der Hochwasserschutzmaßnahmen.

7.1.1 Monokriterielle nominell diskrete Ergebnisse

Für die monokriterielle nominell diskrete Optimierung wurden in den meisten Fällen die Gene Insertion (GI) und der Uniform Crossover (UX) Operatoren verwendet. Es wurde eine

(5+17) Strategie eingesetzt (Notation von Schwefel (1974)). Die Optimierungszeit liegt hier bei etwa 20 Minuten.

7.1.1.1 Reduzierung des Scheitelabflusses am Zielpegel

Das Optimierungsziel ist die Reduzierung des Scheitelabflusses am Zielpegel (ZPG siehe Kapitel 6.4 Abbildung 6-8). Die Zielfunktion lautet:

$$Z = \min Q_{max,Ziel} \quad (7-1)$$

Belastet wurde dabei das System mit den Zuflüssen aus dem Hochwasserereignis des Julis 1984. Hierbei handelt es sich um ein fünfzig bis achtzig-jährliches Ereignis (Schütz, 2007). Die Scheitel der Zuflüsse liegen zwischen $5 \text{ m}^3/\text{s}$ und $46 \text{ m}^3/\text{s}$ (siehe Abbildung 6-3). Um die Funktionsfähigkeit zu erproben, wird als Startkombination P_0 die schlechteste Variante gewählt, nämlich die Nullvariante. Der Abfluss für die Nullvariante ohne den Einfluss der Speicher wurde mit dem Simulationsmodell ermittelt und beträgt am Zielpegel $106 \text{ m}^3/\text{s}$. Der Abfluss am Zielpegel beträgt mit den an der Erft bisher errichteten Hochwasserschutzmaßnahmen (Referenzszenario A) $94,4 \text{ m}^3/\text{s}$. Wird zusätzlich das Hochwasserrückhaltebecken Garsdorf berücksichtigt (Referenzszenario B), beträgt der Abfluss am Zielpegel $81,3 \text{ m}^3/\text{s}$.

In Abbildung 7-1 ist auf der Ordinate „Qmax SE01“ der Maximalabfluss in m^3/s aufgetragen. Der Abflussscheitel kann um $32,3 \text{ m}^3/\text{s}$ auf $73,7 \text{ m}^3/\text{s}$ reduziert werden. Auf der Abszisse ist die Zahl der Simulationen aufgetragen. Der Optimierungslauf beginnt im linken oberen Bereich mit der Startevaluierung, welche aus der schlechtesten Kombination besteht. In diesem Fall bedeutet das, dass für jeden Standort die Nullvariante gesetzt wurde. Alle weiteren Evaluierungen werden von links nach rechts aufgetragen, wobei der Algorithmus mit jeder weiteren Simulation versucht, den Scheitelabfluss am Zielpegel zu minimieren.

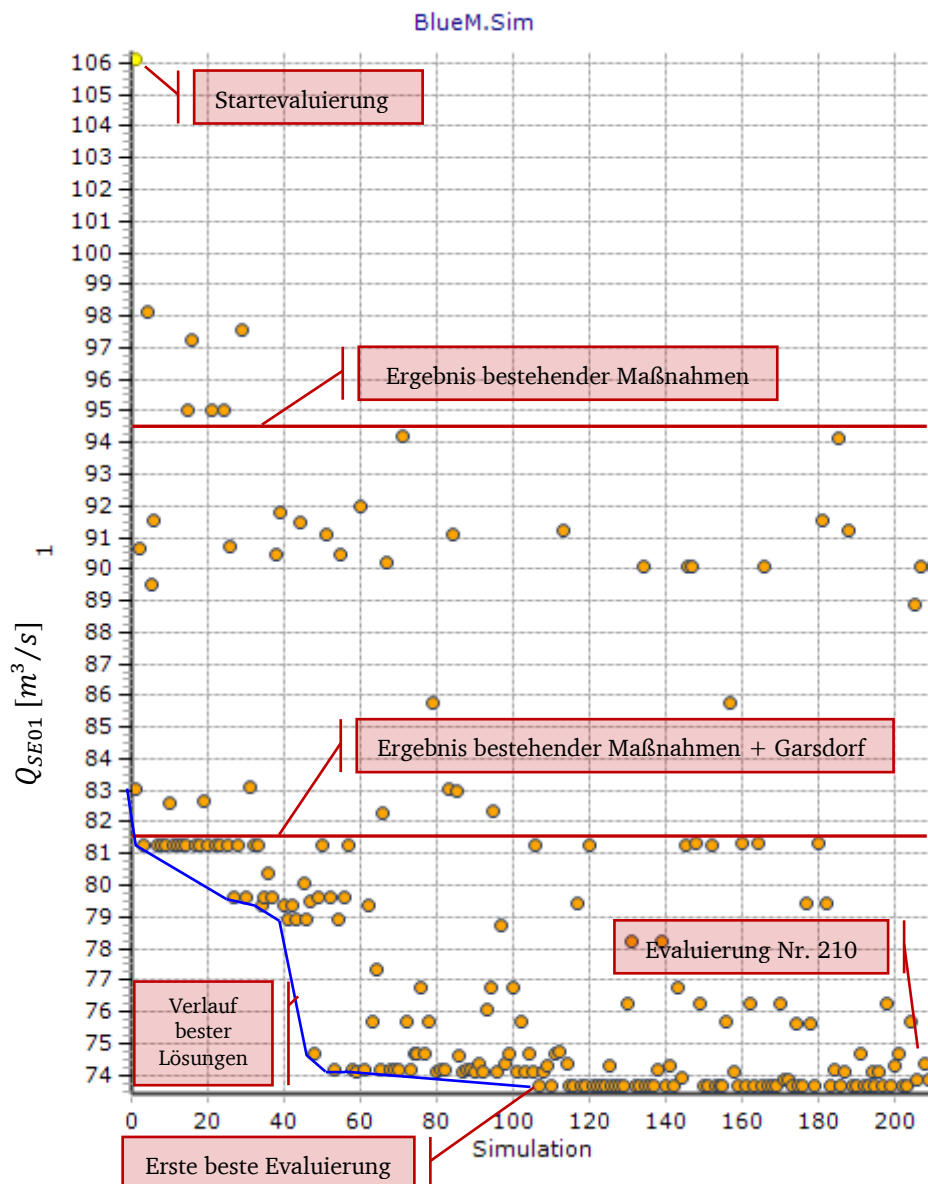


Abbildung 7-1: Monokriterielle nominell diskrete Optimierung mit dem Ziel der Reduzierung des maximalen Abflusses am Zielpegel

Das beste Ergebnis wird nach 108 Simulationen festgestellt und besteht aus folgender Kombination:

1 - 1 - 0 - 2 - 0 - 1 - 1 - 1 - 0 - 2 - 2 - 1 - 1

Diese ermittelte Tour repräsentiert die in Tabelle 7-1 aufgelisteten Maßnahmen für jeden Standort.

Tabelle 7-1: Beste Maßnahmenkombination des monokriteriellen nominell diskreten Optimierungslaufes mit dem Ziel den Scheitelabfluss am Zielpegel zu reduzieren

Nr.	Standort	Tour	Ermittelte Maßnahme
1	Eicherscheid 1	1	HRB Eicherscheid
2	Eicherscheid 2	1	Gerinne Mod
3	Arloff	0	Nullvariante S172
4	Veybach	2	Gerinne, Wiese
5	Horchheim	0	Nullvariante ST03
6	Weilerswist	1	HRB
7	Niederberg	1	HRB Niederberg
8	Moedrath HRB	1	HRB Moedrath
9	Moedrath Gerinne	0	Nullvariante S121
10	Langenich	2	PoldLangenich 1,56
11	Garsdorf	2	HRB Garsdorf 1600
12	Frimmersdorf	1	Füllung Restsee
13	Wevelinghoven	1	Polder

Das erste lokale Optima bei 81,3 m³/s entsteht durch die Aktivierung der Rückhaltebecken Mödrath und Garsdorf. In allen Lösungen dieser Stufe sind die beiden Becken aktiviert. Das zeigt, dass diese Becken einen besonders starken Einfluss auf das Hochwassergeschehen haben, denn sie liegen im Unterlauf der Erft und beeinflussen daher insbesondere den Abfluss am Zielpegel. Die Ergebnisse der bestehenden Maßnahmen (mit und ohne Garsdorf) wurden nachträglich berechnet. Diese beiden Lösungen waren nicht Teil des Gradientenweges. Der folgende Optimierungslauf zeigt, dass die Maßnahmen 1-9 zufällig ausgewählt wurden. Für den Hochwasserschutz am Zielpegel sind lediglich die letzten Becken relevant.

7.1.1.2 Reduzierung der Scheitelabflüsse an mehreren Pegeln durch Aggregation

Das Optimierungsziel ist die Reduzierung des Abflussscheitels Q_{max} im Gesamtsystem der Erft. Betrachtet werden die Abflussspitzen an den Systemelementen:

VV9_Q_zu, SE17_Q_zu, SVEY_Q_zu, SE16_Q_ab, SE14_Q_ab,
S132_Q_ab, SRO1_Q_zu, VV5_Q_zu, VN02_Q_zu, ST11_Q_zu,
SE10_Q_zu, SNE2_Q_ab, SE06_Q_ab, VV2_Q_zu, SE01_Q_ab.

Der maximale Abfluss jeder Position wird mit einem Gewichtungsfaktor belegt, um die unterschiedlichen Volumina zu berücksichtigen. Die Gewichtung der Scheitelabflüsse auf Basis einer Simulation ohne Maßnahmen (Nullvariante) wurde ermittelt, um alle

Positionen gleichwertig zu berücksichtigen. Der größte Scheitelabfluss (Zielpegel) wurde als Normabfluss definiert und mit dem Faktor 1 belegt. Alle anderen Scheitelabflüsse an anderen Positionen wurden mit Faktoren belegt, sodass das jeweilige Produkt dem Normabfluss entspricht. Die Zielfunktion lautet:

$$Z = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-2)$$

Belastet wurde das System ebenfalls mit den Zuflüssen des Hochwasserereignisses des Julis 1984.

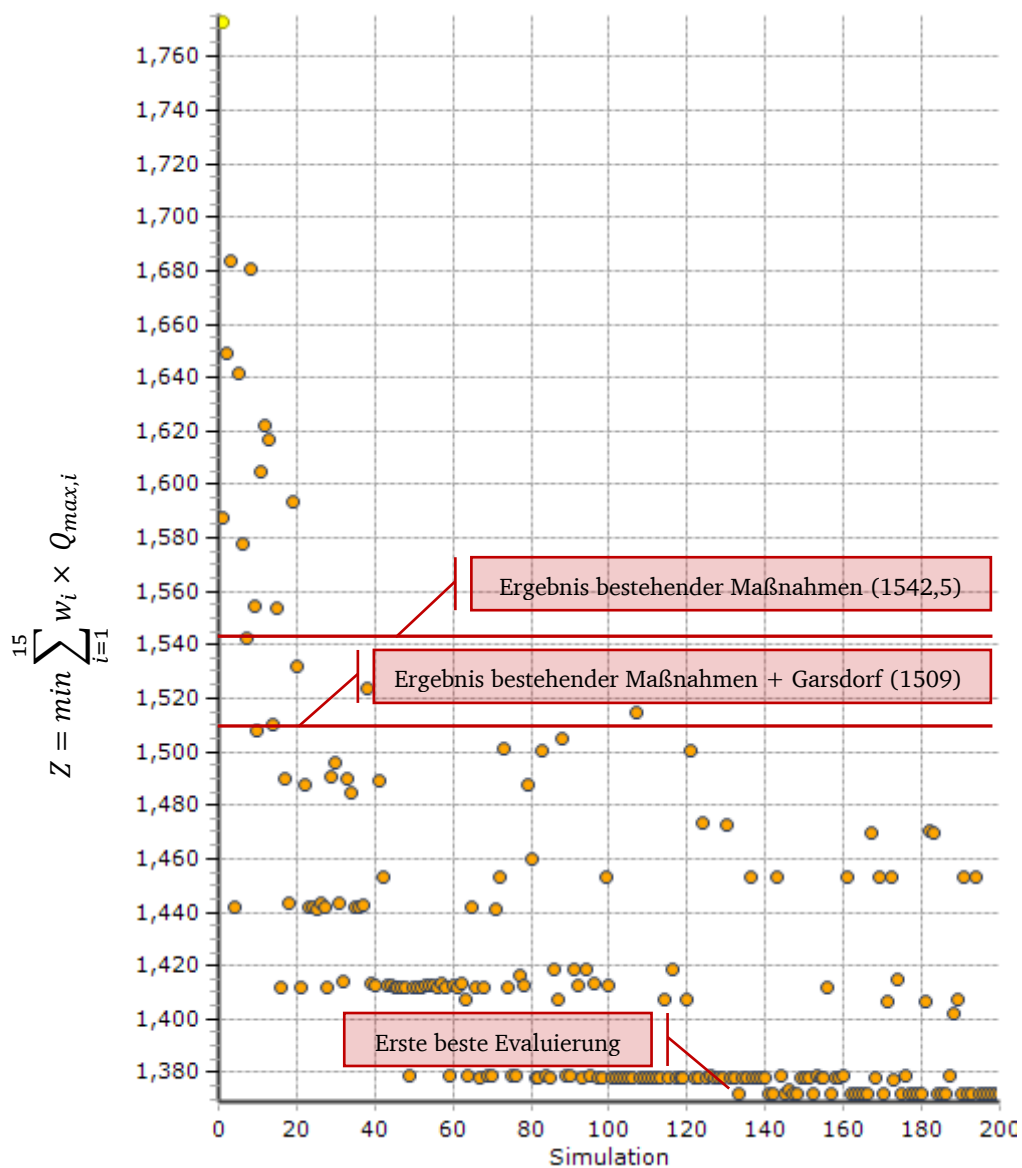


Abbildung 7-2: Optimierungsverlauf zur Reduzierung des Abflussscheitels, Anzahl der Simulationen 200, absolutes Optimum nach 132 Simulationen erreicht.

Der Optimierungsverlauf in Abbildung 7-2 zeigt, dass das absolute Optimum nach 132 Simulationen erreicht wurde. Auch bei weiteren Optimierungsläufen wurde das beste Ergebnis immer etwas später erreicht als bei Optimierungsläufen mit einem Zielpiegel. Zu Beginn der Optimierung nimmt die Qualität schnell zu. Wie zu erwarten verlangsamt sich der Optimierungsfortschritt in der Nähe des Optimums. Das ermittelte Ergebnis ist folgender Pfad:

1 – 2 – 0 – 2 – 1 – 1 – 0 – 1 – 1 – 2 – 2 – 1 – 1

Welche Maßnahmen für die Standorte ermittelt wurden, zeigt die Tabelle 7-2.

Tabelle 7-2: Beste Maßnahmenkombination des monokriteriellen nominell diskreten Optimierungslaufes mit dem Ziel den Scheitelabfluss an mehreren Positionen zu reduzieren

Nr.	Standort	Tour	Tour aus voriger Optimierung	Ermittelte Maßnahme
1	Eicherscheid 1	1	1	HRB Eicherscheid
2	Eicherscheid 2	2	1	HRB Eicherscheid 2
3	Arloff	0	0	Nullvariante S172
4	Veybach	2	2	Gerinne, Wiese
5	Horchheim	1	0	HRB Horchheim
6	Weilerswist	1	1	HRB
7	Niederberg	0	1	Nullvariante SROT
8	Mödrath HRB	1	1	HRB Mödrath
9	Mödrath Gerinne	1	0	Gerinne
10	Langenich	2	2	PoldLangenich 1,56
11	Garsdorf	2	2	HRB Garsdorf 1600
12	Frimmersdorf	1	1	Füllung Restsee
13	Wevelinghoven	1	1	Polder

An 11 von 13 Standorten wurden Maßnahmen aktiviert. An den Standorten Arloff und Niederberg wurden keine Maßnahmen platziert, da diese keinen signifikanten Beitrag zur Reduzierung des Scheitels haben. Im Vergleich zu obiger Optimierung mit einem Zielpiegel unterscheidet sich die Tour vor allen in den oberen beiden Dritteln, während die letzten vier gewählten Maßnahmen identisch sind. Das legt nahe, dass bei obigen Optimierungslauf die Maßnahmen an den Positionen 1 -9 durch die obem gewählte Zielfunktion mit einem Ziel nicht hinreichend berücksichtigt werden. Daher benötigt der Algorithmus auch mehr Simulationen, um eine beste Kombination zu ermitteln.

Im Vergleich zur ersten Optimierung besteht gegenüber den bestehenden Maßnahmen mehr Optimierungspotenzial. Durch das Auswerten der Scheitel an mehreren Positionen, wird den im Oberlauf platzierten Maßnahmen stärker Rechnung getragen. Mit der gewählten aggregierter Zielfunktion wird auch der lokale und regionale Hochwasserschutz berücksichtigt. Natürlich machen diese beiden monokriteriellen Ergebnisse noch keine Aussage über die Qualität der bisherigen Maßnahmen, da es offensichtlich ist, dass einen höhere Zahl an Maßnahmen die maximalen Abflüsse reduziert. Abbildung 7-3 zeigt die beste Maßnahmenkombination im Einzugsgebiet.

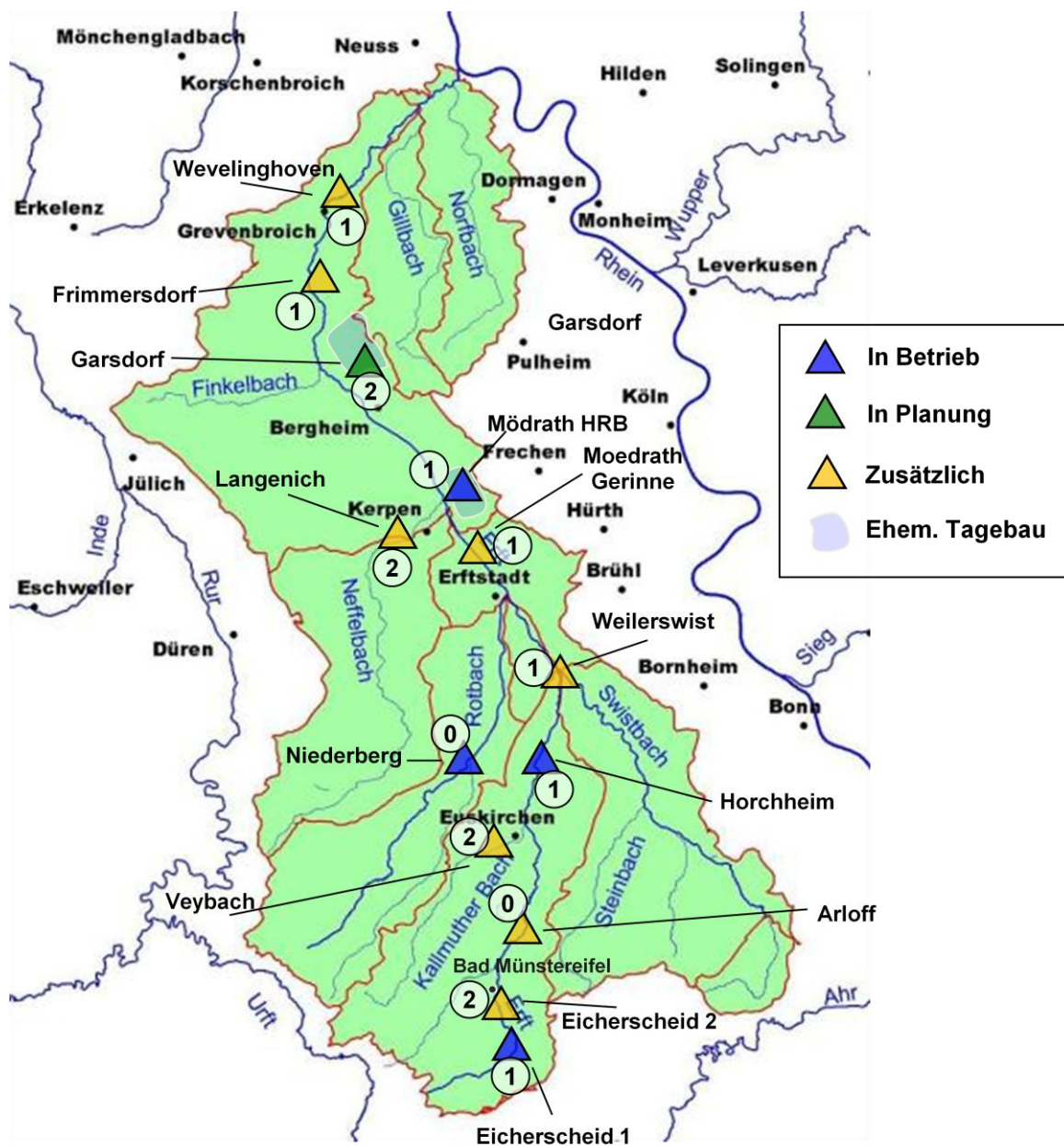


Abbildung 7-3: Beste Maßnahmenkombination bei nominell diskreter Optimierung

7.1.2 Multikriterielle nominell diskrete Ergebnisse

Im folgenden Abschnitt werden multikriterielle nominell diskrete Ergebnisse vorgestellt. Hierbei werden neben den Abflüssen zusätzlich ökologische Faktoren und Investitionskosten für die Maßnahmen berücksichtigt. Zusätzlich wird demonstriert, welchen Einfluss starke Abhängigkeiten zwischen den Zielfunktionen auf die Optimierungsergebnisse haben. Die Paretofront, bestehend aus einem Satz paretooptimaler Lösungen, ist in den Abbildungen mit grünen Punkten dargestellt.

7.1.2.1 Optimierungsziele Hochwasserscheitel und Retentionsvolumina

Für die Optimierung von Speichersystemen unter dem reinen Gesichtspunkt des Hochwasserschutzes kann auf eine Langzeit-Kontinuumssimulation verzichtet werden. Einzelne unabhängige Hochwasserereignisse können in Serie simuliert werden. Möchte man allerdings ökologische Aspekte berücksichtigen, benötigen diese in den meisten Fällen Kontinuumssimulationen, um die Veränderung der hydrologischen Dynamik beurteilen zu können (Richter u. a., 1996). Sollen beispielsweise Indicators of Hydrologic Alteration (IHA) berechnet werden sind Simulationen mit der Zeitspanne von Dekaden erforderlich. Dies würde die hier vorliegenden Anforderungen hinsichtlich der Performanz sprengen. Daher muss auf eine detaillierte ökologische Modellierung verzichtet werden. Stellvertretend wird stark vereinfacht das Retentionsvolumen als Indikator verwendet. Es wird angenommen, dass ein größeres Retentionsvolumen, auch eine größere ökologische Beeinträchtigung darstellt, da dieses Retentionsvolumen durch technische Maßnahmen hergestellt werden muss. Es gilt:

$$\text{Ökologische Beeinträchtigung} \sim \text{Retentionsvolumen} \quad (7-3)$$

Das Optimierungsziel besteht in der Reduzierung der gewichteten Summe der Abflussscheitel Q_{max} und dem Retentionsvolumen V .

$$Z_1 = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-4)$$

$$Z_2 = \min \sum_{i=1}^{15} V_i \quad (7-5)$$

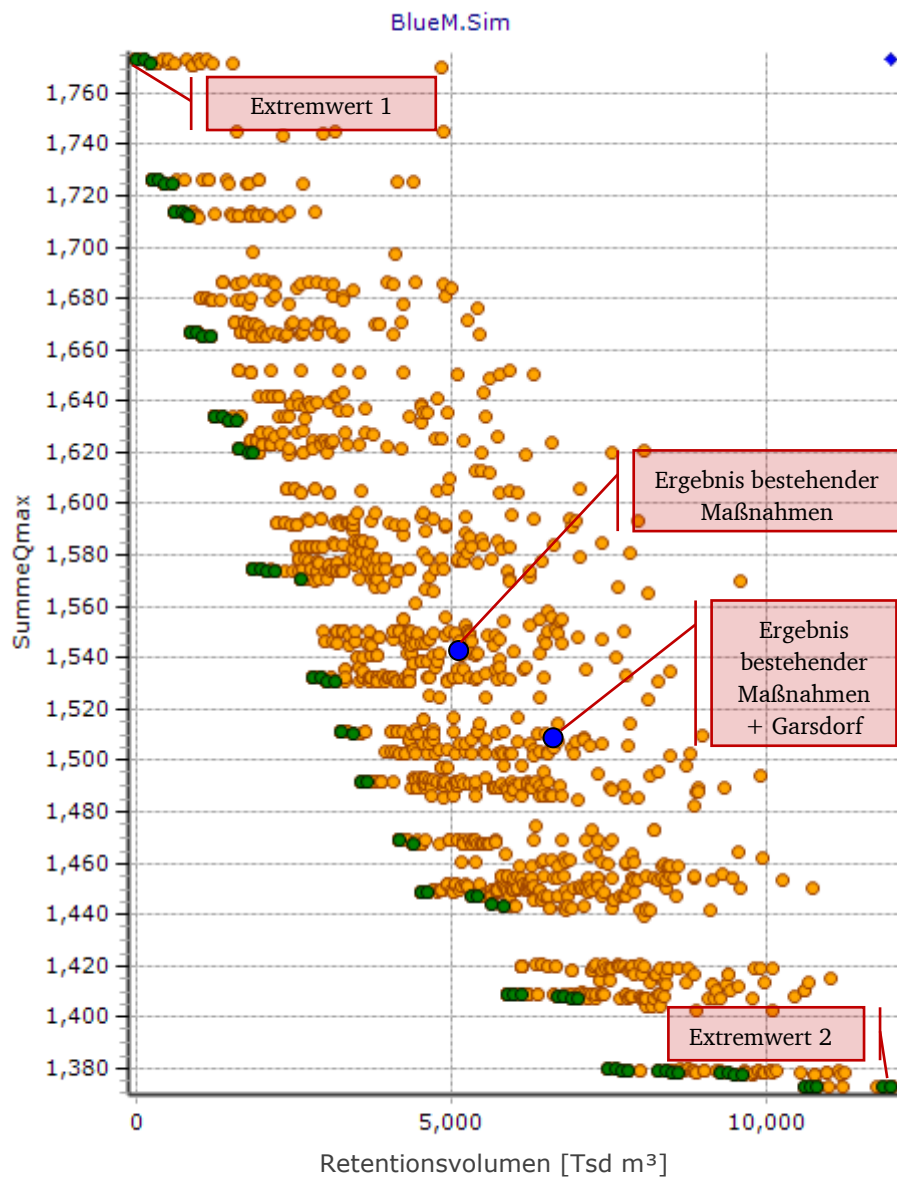


Abbildung 7-4: Multikriterielles nominell diskretes Ergebnis zur Reduzierung des Hochwasserscheitels und des Retentionsvolumens

In Abbildung 7-4 ist die Zielfunktion z_1 zur Reduzierung der Scheitelabflüsse auf der Ordinate aufgetragen und entspricht der gewichteten Summe der Hochwasserscheitel in m^3/s . Die Zielfunktion z_2 zur Reduzierung des Retentionsvolumens in $Tsd.m^3$ ist auf der Abszisse aufgetragen. Die beiden äußeren Ränder der Paretofront sind mit den Extremwerten 1 und 2 gekennzeichnet. Der Extremwert 1 ist eine Lösung, die vollständig aus Nullvarianten besteht. Der Extremwert 2 besteht aus einer Kombination mit Maßnahmen an jedem Standort. Zusätzlich wurden die bestehenden Maßnahmen simuliert und die Ergebnisse eingetragen. Die aktuell realisierten Maßnahmen erreichen für $z_1 = 1542,5 m^3/s$

und für $z_2 = 5.097 \text{ m}^3$. Wird das Hochwasserrückhaltebecken Garsdorf zusätzlich berücksichtigt, beträgt $z_1 = 1.509 \text{ m}^3/\text{s}$ und für $z_2 = 6.697 \text{ m}^3/\text{s}$. Beide Lösungen liegen hinter der Paretofront. Das bedeutet, es existieren Kombinationen aus Speichern, welche entweder den gleichen Schutzgrad bei weniger Volumen oder mit dem gleichen Retentionsvolumen einen besseren Hochwasserschutz ermöglichen. In Tabelle 7-3 sind diese alternativen Lösungen mit A1 und A2 für die bestehenden Maßnahmen und mit B1 und B2 für die bestehenden Maßnahmen und dem HRB-Garsdorf überschrieben.

Tabelle 7-3: Auswahl aus den paretooptimalen Lösungen

		A																B			
		1								2								1		2	
SimulationsNr.		823	296	2030	295	263	560	386	1616	1911	229	826	427	1517	280	318	531	1333	2068		
Z ₁ : Summe Qmax		1773	1726	1713	1667	1633	1621	1574	1532	1511	1491	1468	1449	1409	1379	1374	1372	1372	1372		
Standort	Z ₂ : Retentionsvolumina	0	240	612	852	1252	1624	1864	2812	3329	3564	4277	4512	5977	7577	9777	10573	11823	12388		
	Rang																				
1	HRB Eicherscheid	3	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1		
2	Eicherscheid 2	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
3	Arloff	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
4	Veybach	1	0	2	0	2	2	0	2	2	2	2	2	2	2	2	2	2	2		
5	HRB Horchheim	4	0	0	0	0	0	0	0	1	0	1	0	1	1	1	1	1	1		
6	Weilerswist	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1		
7	HRB Niederberg	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1		
8	HRB Mödrath	5	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1		
9	Mödrath 2	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1		
10	Langenich	2	0	0	1	1	0	1	1	2	1	1	2	2	2	2	2	2	2		
11	HRB Garsdorf	6	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	2	2		
12	Frimmersdorf	7	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1		
13	Wevelinghoven	7	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1		

Die in Tabelle 7-3 blau schattierten Zeilen kennzeichnen die im Einzugsgebiet bestehenden oder geplanten Maßnahmen. Dem Pareto-kriterium entsprechend steigt von links nach rechts die Qualität der Zielfunktion Z_1 , während die Qualität der Zielfunktion Z_2 sinkt. In der dritten Spalte wurde der Rang eingetragen. Dieser berücksichtigt in welcher Reihenfolge die Maßnahmen vom Algorithmus aktiviert wurden, um den Spitzenabfluss Z_1 zu verringern. Anders als in obiger Abbildung 7-4, in der die Resultate der bestehenden Maßnahmen eindeutig hinter der Paretofront liegen, liegen bei dieser Betrachtung die

realisierten Becken weit über dem Durchschnitt. Die Becken Eicherscheid, Horchheim, Mödrath und Garsdorf besetzen Ränge zwischen 3 und 6. Einzig das Becken Niederberg liegt erst auf Rang 10.

Ebenfalls zu beobachten ist, dass die ersten 8 der 18 Lösungen alleine durch die Kombination der Becken Veybach, Langenich und Eicherscheid (Rang 1-3) besetzt sind. Diese müssen bei diesem Ereignis ein besonders gutes Volumen-/Effizienzverhältnis aufweisen.

7.1.2.2 Optimierungsziele Hochwasserscheitel und Investitionskosten

Die Zielgrößen innerhalb des Optimierungsprozesses sind die Reduzierung des Spitzenabflusses und die Reduzierung der Investitionskosten. Der Optimierungsalgorithmus minimiert für die Optimierungsziele Spitzenabfluss und Kosten folgende Ausdrücke:

$$Z_1 = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-6)$$

$$Z_2 = \min \sum_{i=1}^{15} C_i \quad (7-7)$$

$$C = 1269 \times V_R^{0,605} \quad (7-8)$$

Mit: V_R : Zusätzliches Retentionsvolumen

Für die Ermittlung von C wurden im Rahmen einer Recherche 41 Hochwasserrückhaltebecken und deren Investitionskosten zugrunde gelegt. Die Rückhaltebecken Niederberg, Mödrath und Garsdorf im Einzugsgebiet der Erft wurden ebenfalls berücksichtigt. Für Horchheim und Niederberg lagen keine Daten vor. Auf Basis dieser Informationen wurde eine Trendlinie in Form einer Potenzfunktion entwickelt (siehe Abbildung 7-5).

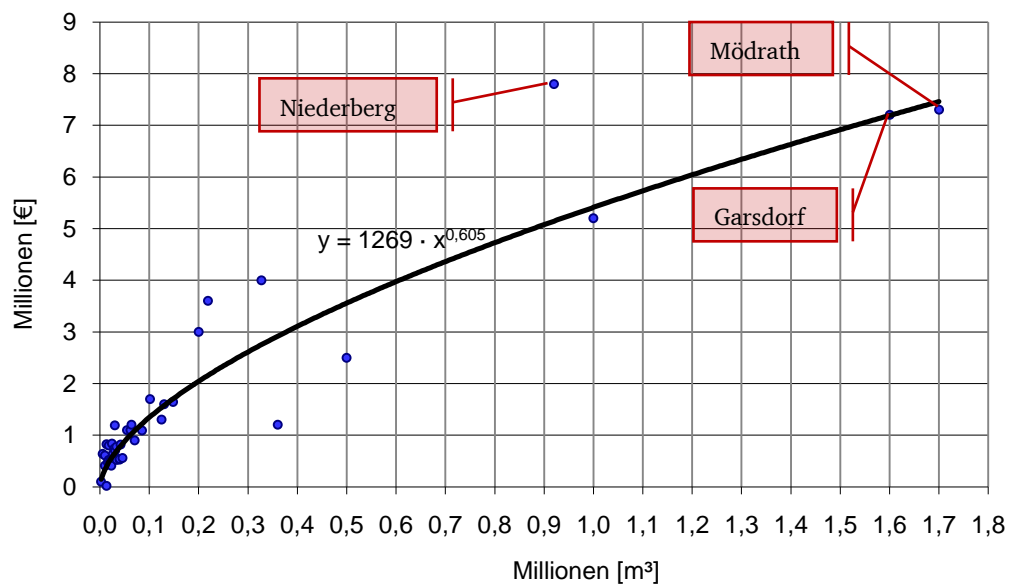
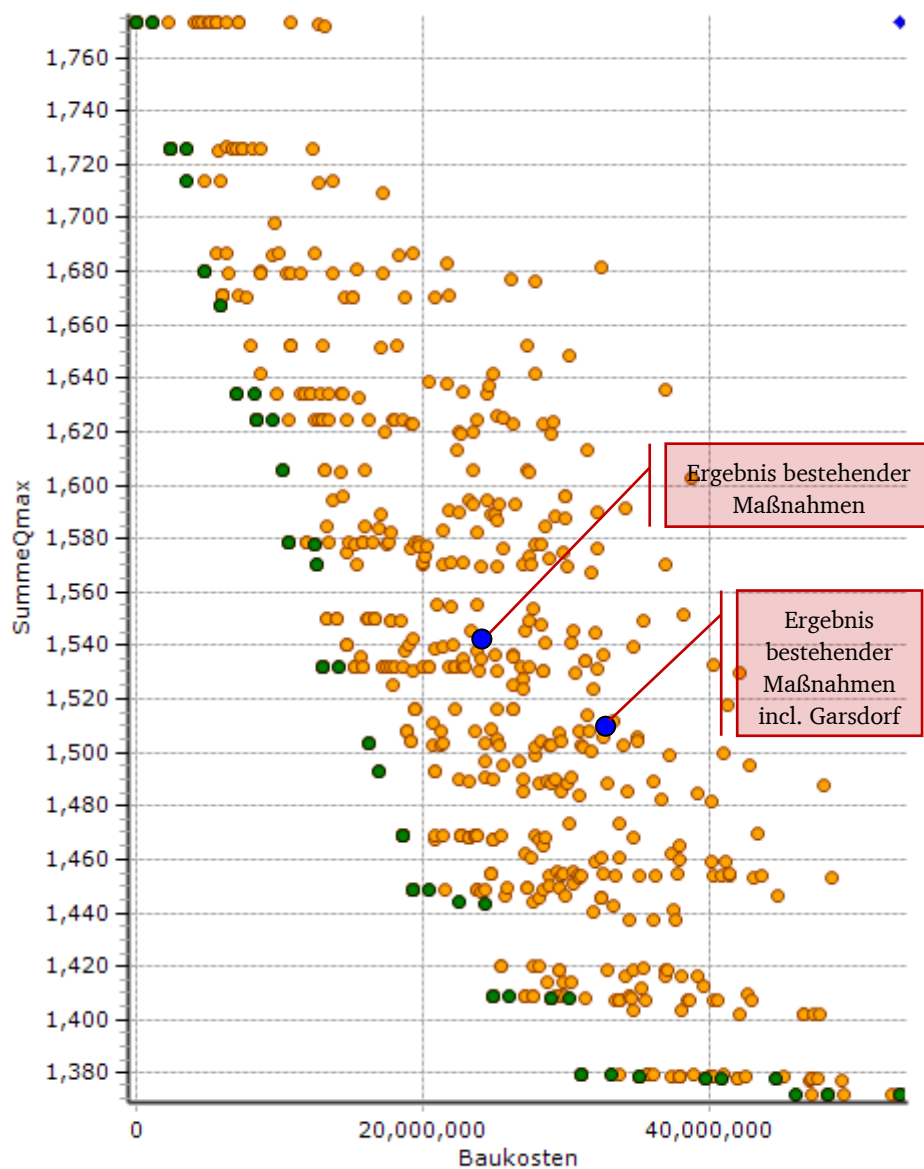


Abbildung 7-5: Datenpunkte und Trendlinie der 41 Hochwasserrückhaltebecken

Die Hochwasserrückhaltebecken Mödrath und Garsdorf liegen relativ exakt auf dieser Potenzialfunktion und das auch dann, wenn die Funktion zunächst ohne die Becken im Einzugsgebiet der Erft entwickelt wurde. Das HRB Niederberg ist im Vergleich zu allen anderen Maßnahmen teurer. Hier schlagen 3,7 Mio. € für Grunderwerb und sonstige Kosten zu Buche. Daher sollte für den Einsatz dieser Potenzialfunktion in der Praxis der Umfang der Stichprobe erweitert werden. Um die Funktionsfähigkeit der in dieser Arbeit entwickelten Algorithmen zu erproben ist diese Potenzialfunktion aber hinreichend.



Die aktuell realisierten Maßnahmen erreichen für $Z_1 = 1.542,5 \text{ m}^3/\text{s}$ und für $Z_2 = 24,88 \text{ Mio €}$. Wird das Hochwasserrückhaltebecken Garsdorf zusätzlich berücksichtigt, beträgt $Z_1 = 1.509 \text{ m}^3/\text{s}$ und für $Z_2 = 32,07 \text{ Mio €}$. Beide Lösungen liegen erneut hinter der Paretofront.

Tabelle 7-4: Auswahl aus den paretooptimalen Lösungen

Simulationsnr.			839	648	559	1097	508	634	74	1805	405	293	661	512	367	454	587	1553	1543	2070
Z ₁ : Summe Qmax			1773	1726	1714	1680	1667	1634	1621	1575	1532	1512	1492	1469	1449	1409	1380	1374	1373	1373
Standort	Z ₂ : Investitionskosten [Mio. €]	Rang	0,00	2,28	4,02	5,45	6,31	7,74	9,47	11,75	14,82	18,57	19,22	21,64	22,28	29,10	36,29	50,26	61,21	64,32
	1	HRB Eicherscheid	3	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1
2	Eicherscheid 2	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	1
3	Arloff	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	Veybach	1	0	2	0	0	2	2	0	2	2	2	2	2	2	2	2	2	2	2
5	HRB Horchheim	4	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1	1	1
6	Weilerswist	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
7	HRB Niederberg	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
8	HRB Mödrath	5	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1
9	Moedrath 2	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
10	Langenich	2	0	0	1	0	1	0	1	1	2	1	1	2	2	2	2	2	2	2
11	HRB Garsdorf	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	2
12	Frimmersdorf	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
13	Wevelinghoven	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

In Tabelle 7-4 werden in blau die Felder markiert, die sich von obigen Optimierungsergebnissen in Tabelle 7-3 unterscheiden. Es wird deutlich, dass durch die hinterlegte Potenzialfunktion für die Investitionskosten kaum Änderungen bestehen. Die meisten Maßnahmen werden etwas früher oder später aktiviert. Auch der Rang der Maßnahmen ist nahezu identisch. Die einzigen Ausnahmen bilden Eicherscheid 2 und Weilerswist. Eicherscheid 2 hat hier den Rang 7 statt 10 und Weilerswist den Rang 10 statt 9.

7.1.2.3 Optimierungsziel Reduzierung des maximalen Scheitelabflusses bei zwei abhängigen Zielen

Für den Erfolg zur Ermittlung einer Paretofront ist erstens die Unabhängigkeit der Zielsetzungen entscheidend und zweitens sollten es konkurrierende Zielsetzungen sein. Sind Zielsetzungen nicht vollständig unabhängig, kann es von Vorteil sein, monokriteriell statt multikriteriell zu optimieren und das zweite Ziel mittels einer Abbildungsfunktion zu ermitteln. Mit den folgenden Beispielen sollen die Anwendungsbereiche der multikriteriellen Optimierung verdeutlicht werden. Das Erste zeigt die Optimierung des maximalen Scheitelabflusses auf zwei Ziele aggregiert. In der Zielfunktion 1 werden die Scheitel-

abflüsse der Positionen 1-7 und in der Zielfunktion 2 die Scheitelabflüsse der Positionen 1-15 im Einzugsgebiet der Erft optimiert.

$$Z_1 = \min \sum_{i=1}^7 w_i \times Q_{max,i} \quad (7-9)$$

$$Z_2 = \min \sum_{i=8}^{15} w_i \times Q_{max,i} \quad (7-10)$$

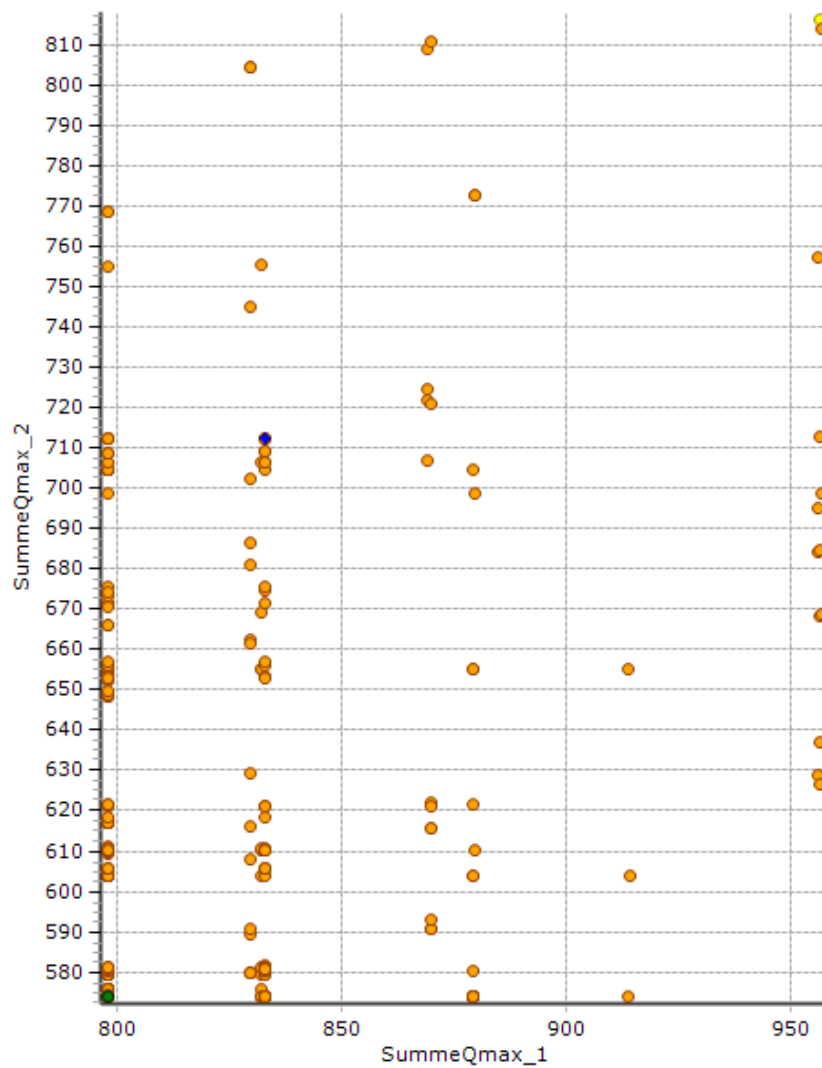


Abbildung 7-6: Optimierung zweier nicht konkurrierender Ziele

Da diese beiden Ziele nicht konkurrierend sind, ist das Ergebnis, wie in Abbildung 7-6 ersichtlich, nicht ein Satz von Lösungen in Form einer Paretofront, sondern ein einzelnes

Optimum. Das Ergebnis ist zwar nicht „falsch“, es hätte aber auch ohne einen multi-kriteriellen Ansatz ermittelt werden können.

In manchen Fällen sind die Ziele zwar konkurrierend, aber es ist trotzdem nicht erforderlich, multikriteriell zu optimieren. Dennoch kann die ermittelte Paretofront einen Hinweis auf einen bestehenden funktionalen Zusammenhang liefern. Das zweite Beispiel aus der Arbeit Sharma (2008) zeigt einen eindeutig bereichsweise funktionalen Zusammenhang zwischen den Zielfunktionen. Im linken Ast in der Abbildung 7-7 ist ein linearer Zusammenhang zwischen den Zielfunktionen f_1 und f_2 bis im Bereich f_1 : $-2.0e+7$ bis $-5.0e+6$ zu erkennen. Im Bereich $f_2 > -0,75e+6$ wird der Zusammenhang durch eine nicht lineare Funktion abgebildet.

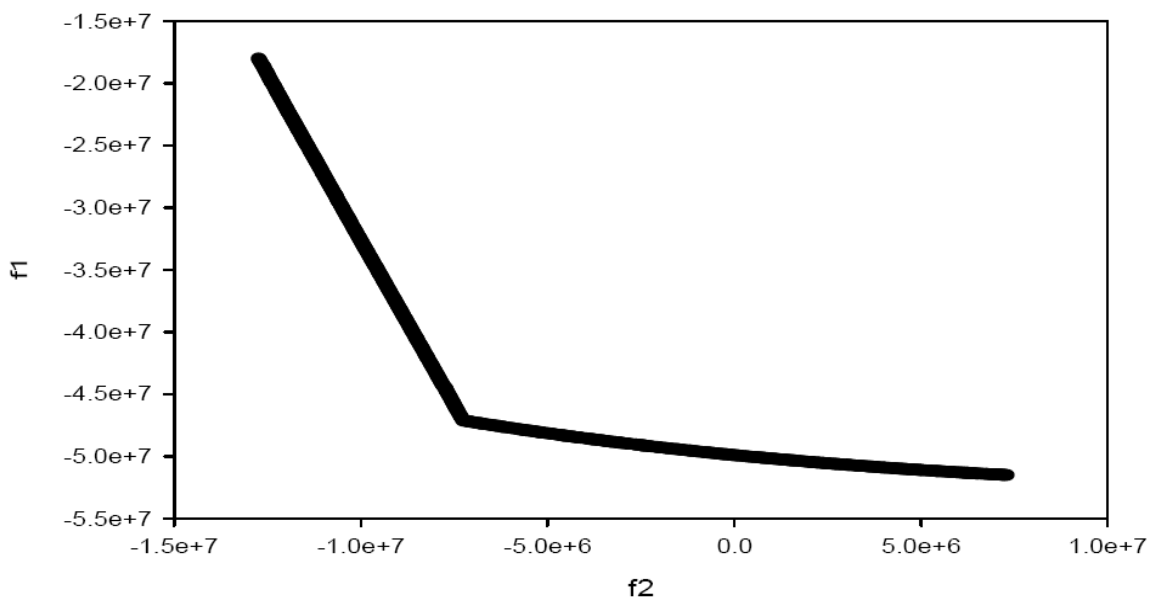


Abbildung 7-7: Beispiel eines eindeutig bereichsweise funktionalen Zusammenhangs zwischen den Zielfunktionen

In diesem Beispiel wurde die Energiegewinnung der Urftalsperre an der Rur optimiert und gleichzeitig versucht, möglichst viel Retentionsvolumen für kommende Hochwasser zu erhalten. Eingangsinformation war eine Abflussprognose, die mithilfe eines neuronalen Netzes erstellt wurde. Das für die Optimierung verwendete Modell beruhte auf wenigen Gleichungen. Gegebenenfalls hätten diese Zusammenhänge durch Ableiten und Umformen der Gleichungen ermittelt werden können. Daher sollte vor der evolutionären Optimierung

stets untersucht werden, ob es erforderlich ist, derartig rechenintensive Algorithmen einzusetzen.

7.2 Optimierungsergebnisse des memetischen Algorithmus

Bei der vorher beschriebenen nominell diskreten Optimierung bleiben die reellwertigen Parameter der einzelnen Maßnahmen unverändert. Der entwickelte memetische Algorithmus erlaubt es, neben den Kombinationen auch die reellen Parameter zu optimieren. Für eine erfolgreiche memetische Optimierung ist es erforderlich, die reellen und nominellen Operatoren mit Hilfe ihrer Steuerparameter so einzustellen, dass sie sich nicht gegenseitig behindern, sondern ihre Stärken ausspielen können und Synergien nutzen können.

7.2.1 Monokriterielle Ergebnisse des memetischen Algorithmus

Für die Optimierung wurde die Methode der parallelen Optimierung mit vorselektierten reellen Parametern eingesetzt (siehe Kapitel 3.2.4.3 Seite 66). Für die nominelle Reproduktion wurde der Uniform Crossover Operator und für die Mutation der Random Bit Value Operator verwendet. Für die Mutation erwies sich eine Rate zwischen 5 % und 10 % als praktikabel. Beim Memory Selektionsoperator wurde die Einstellung One-Location-Up verwendet. Für die reellwertige Reproduktion wurde eine diskrete Rekombination verwendet. Dabei werden diskret bernulliverteilt die reellen Werte des Elters A oder des Elters B vererbt. Für die reelle Mutation wurde der Mutationsoperator von Rechenberg verwendet. Dessen Startschrittweite wurde auf 0,04 gesetzt. Mit dieser relative kleinen Startschrittweite haben die reellen Parameter genug Zeit um sich langsam anzupassen, um die Unterbrechung durch die nominelle Mutation abubrechen. Es wurde nominell diskret mit einer $\mu_c = 3$ und $\lambda_c = 7$ Strategie optimiert der Selektionsdruck liegt demzufolge bei 0,43, also etwas geringer als die von Rechenberg vorgeschlagenen Werte zwischen 0,33 und 0,2. Dies erweis sich als vorteilig bei der memetischen Optimierung, da mit dem Selektionsoperator E_{mem} erneut selektiert werden muss. Für die reelle Selektion wurde $\mu_r = 3$ eingesetzt. Der Wert für λ muss von der nominellen Selektion übernommen werden, da $\lambda_r = \lambda_c$. Der reelle Selektionsdruck beträgt demzufolge μ_r/λ_c und kann „lediglich“ über die Anzahl der Eltern μ_r gesteuert werden.

Da bei der memetischen Optimierung zweifach selektiert wird kann durch Multiplikation der beiden Selektionsraten μ_c/λ_c und μ_r/λ_r mit

Memetische
Selektionsrate:

$$S_{mem} = \frac{\mu_r}{\lambda_r} \cdot \frac{\mu_c}{\lambda_c} \quad (7-11)$$

Berechnet werden. Setzt man obige Werte für in Gleichung (7-12) ein erhält man die im folgenden Beispiel angewandte memetische Selektionsrate von 0,185. Mit $0,185 \cong 0,2$ liegt Selektionsrate in dem von Rechenberg geforderten Bereich.

Die erforderliche Rechenzeit für die monokriterielle memetische Optimierung lag bei circa 60 Minuten.

7.2.1.1 Reduzierung der Scheitelabflüsse an mehreren Pegeln durch Aggregation

Das Optimierungsziel entspricht dem in Kapitel 7.1.1.2 „Reduzierung der Scheitelabflüsse an mehreren Pegeln durch Aggregation“. Die Zielfunktion lautet:

$$z = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-12)$$

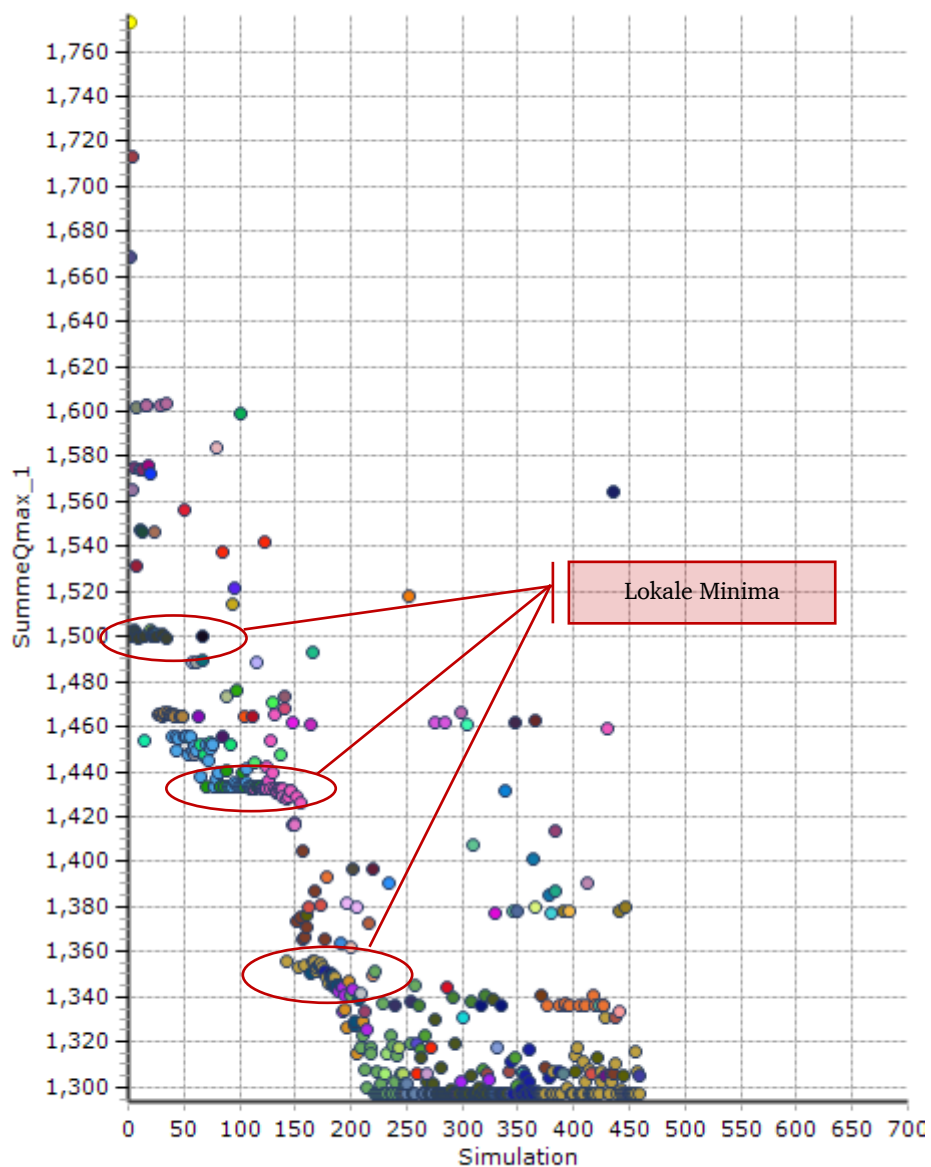


Abbildung 7-8: MEA Optimierungsverlauf mit einer Zielsetzung

Die Qualität der Lösung konnte im Vergleich zur nominellen Optimierung von 1.380 auf unter 1.300 verbessert werden. Deutlich zu erkennen ist der im Vergleich zu einer rein reellwertigen Optimierung stärker ausgeprägte stufenförmige Verlauf. Es sind drei gut ausgebildete Stufen bei 1.500, 1.430 und 1.350 zu erkennen. Hierbei handelt es sich um die bereits erwähnten lokalen Minima. Jede Kombination von Maßnahmen ist prinzipiell ein lokales Minimum.

Der Populationsspeicher P_N des nominell diskreten Algorithmus beinhaltet zunächst nur verschiedene Kombinationen. Erweist sich eine Kombination als qualitativ, so nimmt der

Populationsspeicher P_N mehrere Individuen dieser Kombination mit unterschiedlichen reellen Parametern auf. In dieser Phase werden die reellwertigen Parameter dieser Kombination verbessert. Die geringe Mutationswahrscheinlichkeit von 5 % - 10 % erlaubt den nominellen Operatoren zum einen mehrere Individuen der gleichen Kombination in den Elternspeicher aufzunehmen und zum anderen die reellwertige Anpassung der Parameter. Die Mutationswahrscheinlichkeit ist hier die Wahrscheinlichkeit, dass nach der Reproduktion jedes Allel verändert wird. Eine Mutationswahrscheinlichkeit größer 10 % unterbricht den reellwertigen Algorithmus bei der Parameteranpassung. Ist die Mutationswahrscheinlichkeit kleiner als 5 % läuft der Optimierungsprozess stärker Gefahr, an einem dieser lokalen Minima zu verharren und das globale Optimum nicht zu finden. Die Mutationsrate hat sich als besonders sensitiv für eine erfolgreiche memetische Optimierung erwiesen. Dies bestätigen die Ergebnisse von Sharif und Wardlaw (2000), die ebenfalls eine besondere Sensitivität des Parameters Mutationswahrscheinlichkeit beschreiben. Generell muss bei dieser Art der Optimierung verhindert werden, dass die reellwertige Optimierung nicht durch die nominelle Optimierung überschattet wird.

Um dieses Risiko zu reduzieren, im lokalen Minimum stecken zu bleiben, müssen mithilfe der nominellen Mutation weiterhin neue bisher nicht erprobte Kombinationen evaluiert werden. Dies führt zu den Ergebnissen weit oberhalb der Linie aktuell guter Ergebnisse. Diese sind erforderlich, um die zwischenzeitlich durch die reellwertige Optimierung veränderten Parameter in Verbindung mit anderen Kombinationen hinsichtlich ihrer Qualität zu überprüfen.

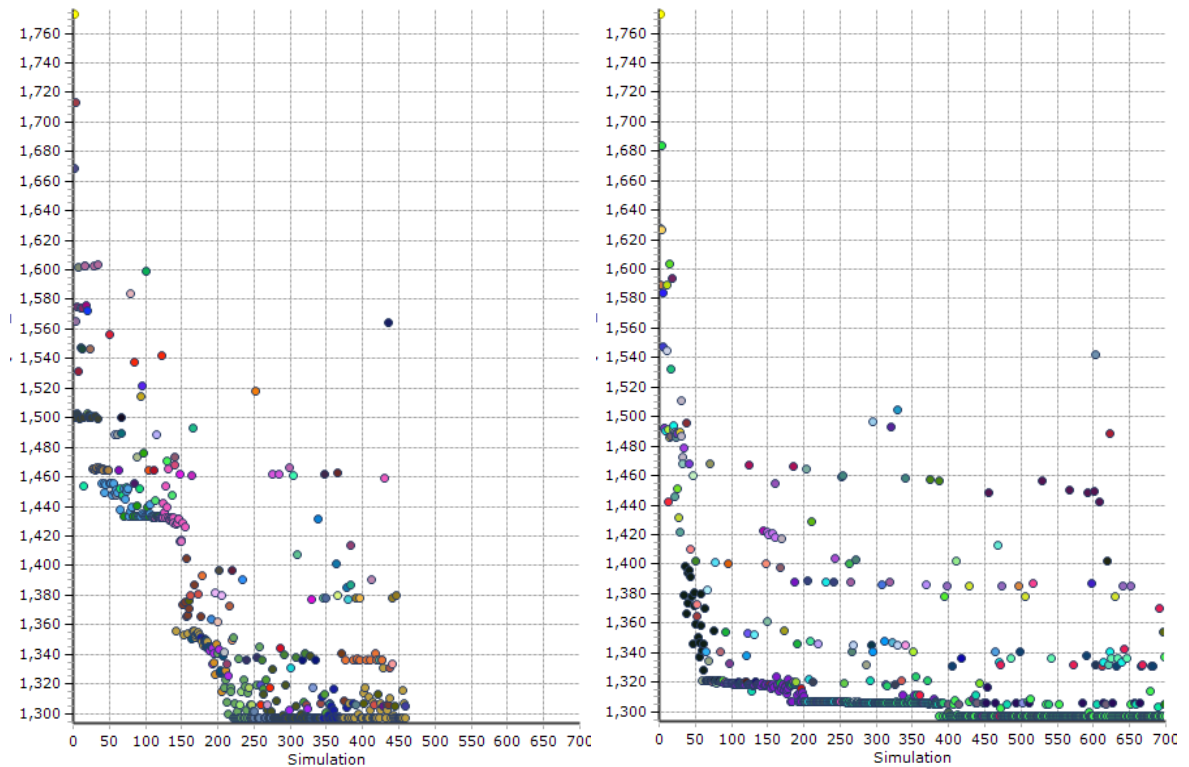


Abbildung 7-9: Vergleich der Gradientenwege bei monokriteriellen MEA Optimierungsverläufen.

Bei der evolutionären Optimierung handelt es sich stets um einen stochastischen Prozess. Selbst bei exakt gleichen Startbedingungen wird der Weg durch das Gebirge des Lösungsraumes immer ein anderer sein. Auch die gefundene Lösung ist nicht zwingend dieselbe. In Abbildung 7-9 wurde in beiden Fällen zwar die gleiche Lösung gefunden, aber der von Rechenberg (1973, 1994) beschriebene Gradientenweg zu dieser Lösung war unterschiedlich. Auf der linken Seite ist der bereits in obiger Abbildung 7-8 dargestellte Optimierungsverlauf zu sehen. Rechts daneben ein anderer Optimierungsverlauf bei gleichen Startbedingungen. Während der linke Verlauf bei einem Qualitätswert von 1.430 in einem lokalen Minimum verharrt, überspringt der rechte Optimierungsverlauf diese Marke, verharrt dann aber bei den Qualitätswerten 1.320 sowie 1.310 und erreicht erst nach ca. 380 Simulationen die beste Lösung. Die beiden Optimierungsläufe unterscheiden sich hinsichtlich der Dominanz der Algorithmen. Beim linken Optimierungslauf konnten die reellwertigen Parameter während des Verharrens auf den lokalen Minima angepasst werden. Den rechten Verlauf kennzeichnet eine anfänglich gute Wahl der Kombination, welche dazu führt, dass die Qualität zu Beginn schnell steigt. Im weiteren Verlauf fehlt

diesem aber die Information für das Adaptieren der reellwertigen Parameter, sodass das beste Ergebnis erst nach 380 statt nach 210 Simulationen ermittelt werden konnte.

7.2.1.2 Reduzierung der Scheitelabflüsse an mehreren Pegeln mit Optimierung der konstanten Abgaben

In diesem Beispiel wurden die Lage, die Volumina und die konstanten Abgaben optimiert. Die Zielsetzung bleibt dabei unverändert Reduzierung der Scheitelabflüsse an mehreren Pegeln. Abbildung 7-10 zeigt, dass bei zusätzlicher Berücksichtigung der Abgaben entsprechend besser Qualitätswerte zu erzielen sind.

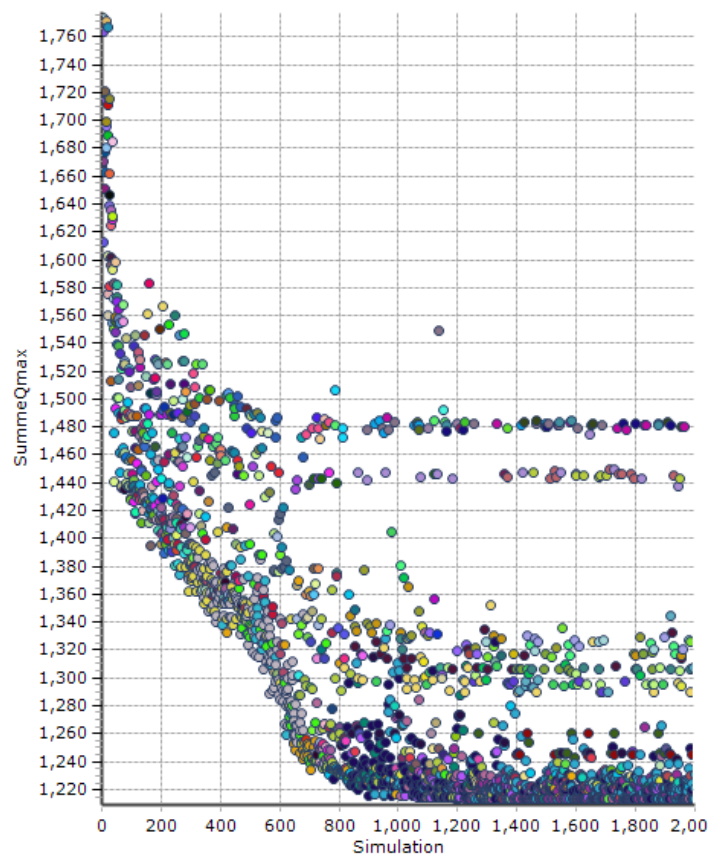


Abbildung 7-10: Memetisches Ergebnis bei der Optimierung Lage, der Volumina und der Abgaben

Das Optimum wurde nach circa 1200 Simulationen erreicht. Die folgende Abbildung 7-11 zeigt zunächst die Abflüsse ohne die Beeinflussung durch Maßnahmen.

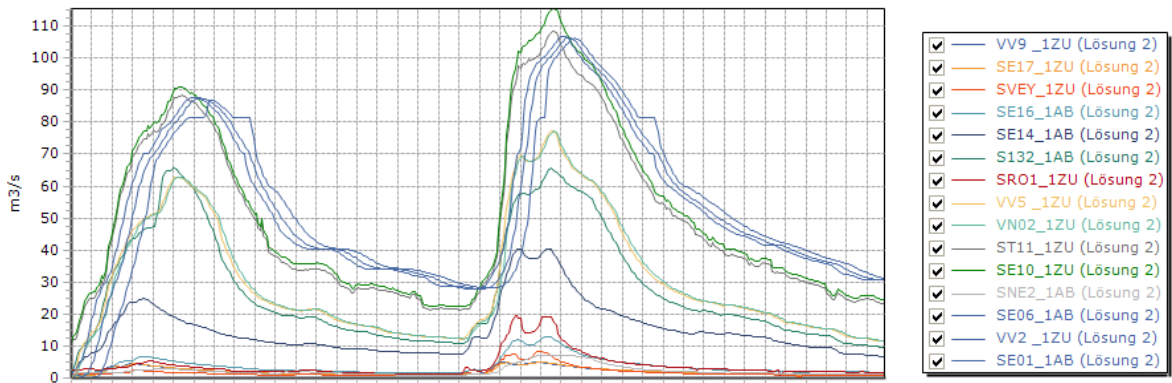


Abbildung 7-11: Abflüsse ohne Maßnahmen

Durch den Einsatz und der Optimierung der Lagen, der Volumina und der konstanten Abgaben können die Abflüsse um 45 m³/s reduziert werden (Abbildung 7-12).

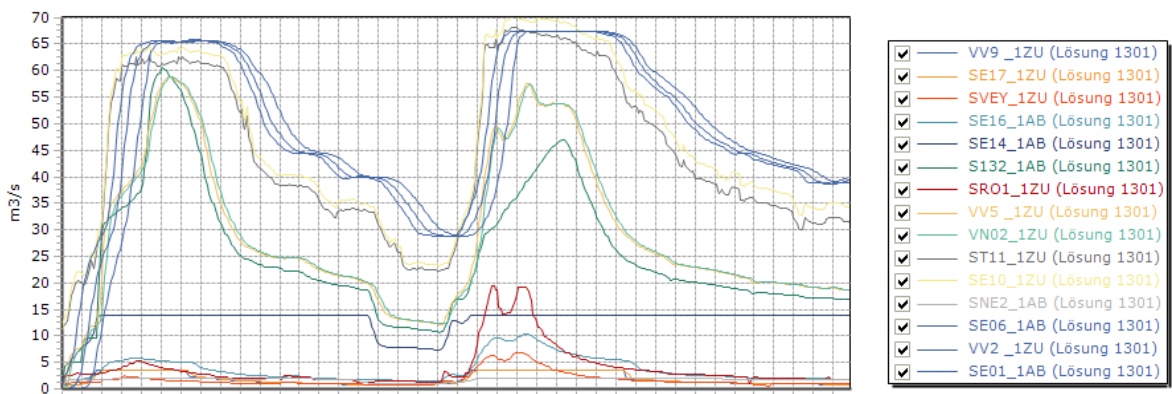


Abbildung 7-12: Abflüsse in Folge der besten Lösung

Zum Vergleich die Abflüsse in Folge der Referenzkombination von Maßnahmen bestehend aus den Becken Eicherscheid, Horchheim, Niederberg, Mödrath dem geplanten Becken Garsdorf (Abbildung 7-13).

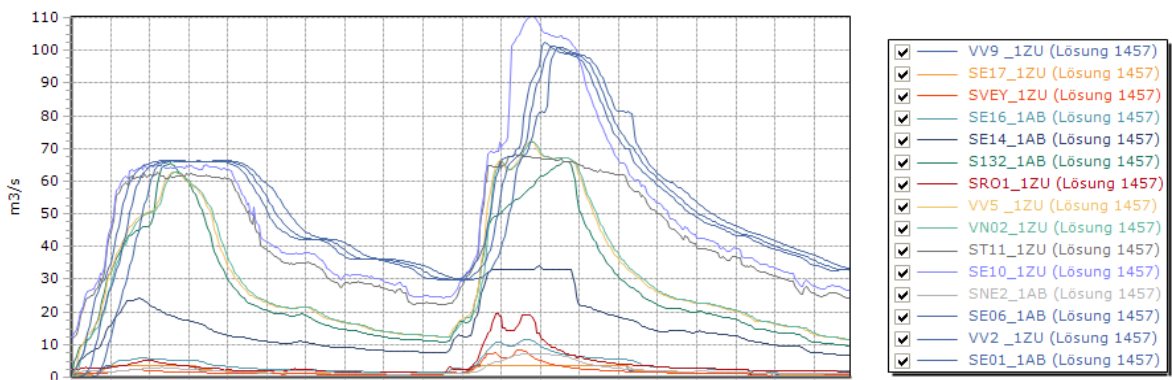


Abbildung 7-13: Abflüsse in Folge der Referenzkombination

Mit der Referenzkombination werden die Abflüsse nur unwesentlich reduziert, obwohl die konstanten Abgaben für den besseren Vergleich rein reellwertig optimiert wurden. Bei diesem voluminösen Hochwasserereignis sind die vorhandenen Speichervolumina zu klein um die Scheitel merklich zu reduzieren. Es handelt sich um ein Ereignis mit einem Wiederkehrintervall von circa siebzig Jahren (Schütz, 2007). Daher müssen also wesentlich mehr Rückhaltevolumina geschaffen werden, um das gesetzte Ziel des Hochwasserschutzes bei einem Ereignis mit einem Wiederkehrintervall von einhundert Jahren zu erreichen.

7.2.2 Multikriterielle Ergebnisse des memetischen Algorithmus

Im folgenden Abschnitt werden multikriterielle memetische Optimierungsergebnisse vorgestellt. Seitens der nominellen Mutation von ist im multikriteriellen Fall eine Rate bis zu 15 % geeignet. Damit werden die Ränder der Paretofront schneller erfasst. Die reelle Mutation erfolgt erneut mit dem Mutationsoperator von Rechenberg, allerdings mit einer etwa höheren Startschrittweite von 0,08. Für die Mono- also auch für die Multikriterielle memetische Optimierung wurde zur reellen Mutation auch der Operator von Schwefel verwendet. Dieser war für die memetische Optimierung völlig ungeeignet. Der Algorithmus begann nicht zu konvergieren, was wahrscheinlich an den im Vergleich zum Operator von Rechenberg an den großen Mutationsschritten lag.

Bei der multikriteriellen Optimierung betrug die Anzahl der Eltern $\mu_c = 30$ und der Nachfahren $\lambda_c = 100$. Der reelle Selektionsdruck wurde mit $\mu_r = 15$ verdoppelt. Da im multikriteriellen Fall auch wesentlich mehr Individuen im Populationsspeicher P_{mem} vorhanden sind kann hier auch stärker selektiert werden. Die Erfahrungen mit P_{mem} haben gezeigt, dass dieser bei der $5 \cdot \mu$ bis $10 \cdot \mu$ fachen Größe einer Generation liegen sollte. Die Untergrenze bedingt der memetische Algorithmus. Ist $P_{mem} < (5 \cdot \mu)$ enthält er zu wenige Individuen die für die Selektion zur Verfügung stehen. Die obere Grenze hat einen anderen Grund. Bei einer memetischen Optimierung muss die $E_{mem,M}$ Selektion wie bereits erwähnt für jedes Allel ausgeführt werden. Bei der multikriteriellen memetischen Optimierung muss daher das Non-Dominated-Sorting n (Anzahl der Standorte) mal ausgeführt werden. Das Non-Dominated-Sorting ist nach der Evaluierung mit dem Simulationsmodell der Prozess im Optimierungsalgorithmus, der am meisten Rechenzeit benötigt, da die Qualitäten aller Individuen mehrfach verglichen werden müssen. Je größer P_{mem} desto mehr Individuen müssen miteinander verglichen werden. Bei der memetischen Optimierung ist dieser

Prozess so zeitintensiv, dass er bei einer sehr großen Populationsgröße P_{mem} relevant wird. Daher wurde in den folgenden Beispielen P_{mem} auf 200 gesetzt.

Die erforderliche Rechenzeit für die multikriterielle memetische Optimierung lag beim ersten Beispiel mit zwei Zielfunktionen bei circa drei Stunden und bei der Optimierung mit vier Zielfunktionen bei circa 15 Stunden.

7.2.2.1 Reduzierung des Hochwasserscheitels, des Retentionsvolumens und der Investitionskosten

Die oben dargestellten multikriteriellen Optimierungsläufe wurden hier zusammengefasst, da durch die zusätzliche Potenzialfunktion für die Investitionskosten nur wenige Unterschiede bestehen. Die Zielfunktionen für die Reduzierung des Hochwasserscheitel und des Retentionsvolumens lauten:

$$z_1 = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-13)$$

$$z_2 = \min \sum_{i=1}^{15} V_i \quad (7-14)$$

Für die Reduzierung des Hochwasserscheitels und der Investitionskosten lauten die Zielfunktionen:

$$z_1 = \min \sum_{i=1}^{15} w_i \times Q_{max,i} \quad (7-15)$$

$$z_2 = \min \sum_{i=1}^{15} C_i \quad (7-16)$$

Bei der Abbildung 7-14 und der Abbildung 7-15 wurden die Abszissen skaliert.

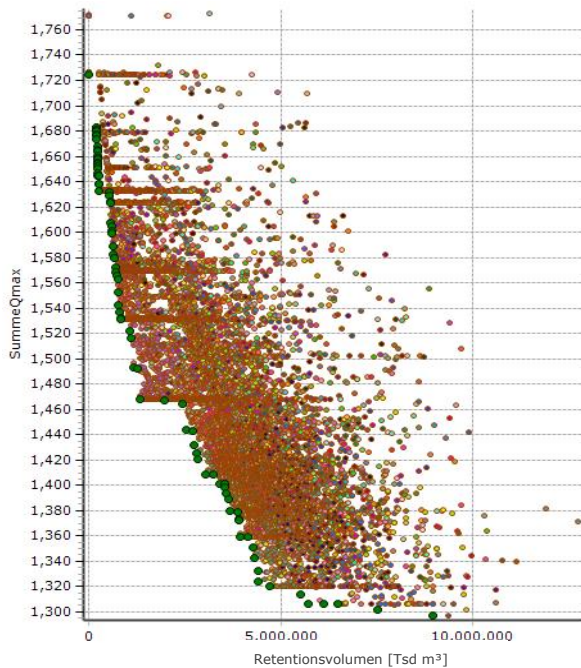


Abbildung 7-14: MEA Optimierung des HW-Scheitels und des Retentionsvolumens

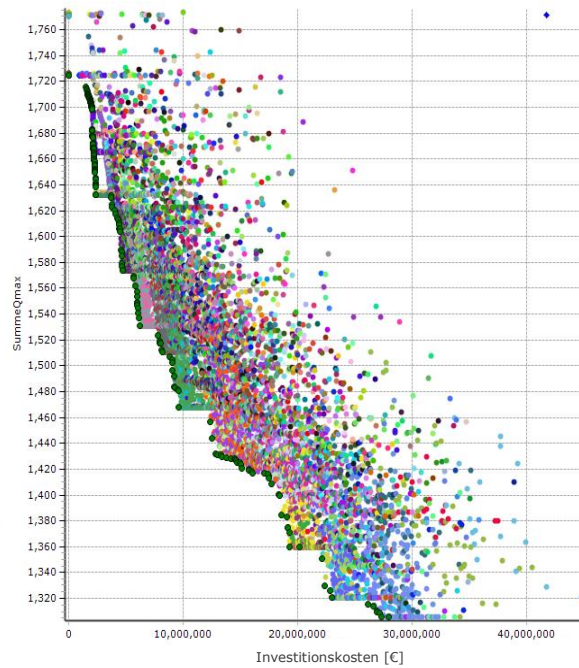


Abbildung 7-15: MEA Optimierung des HW-Scheitels und der Investitionskosten

Der Einfluss der Potenzialfunktion bleibt an den maximalen Werten auf der Abszisse erkennbar. Die Paretofront bei der Kostenoptimierung steigt daher bei kleiner werdenden Werten auf der Abszisse weniger stark an als die Paretofront des Retentionsvolumens.

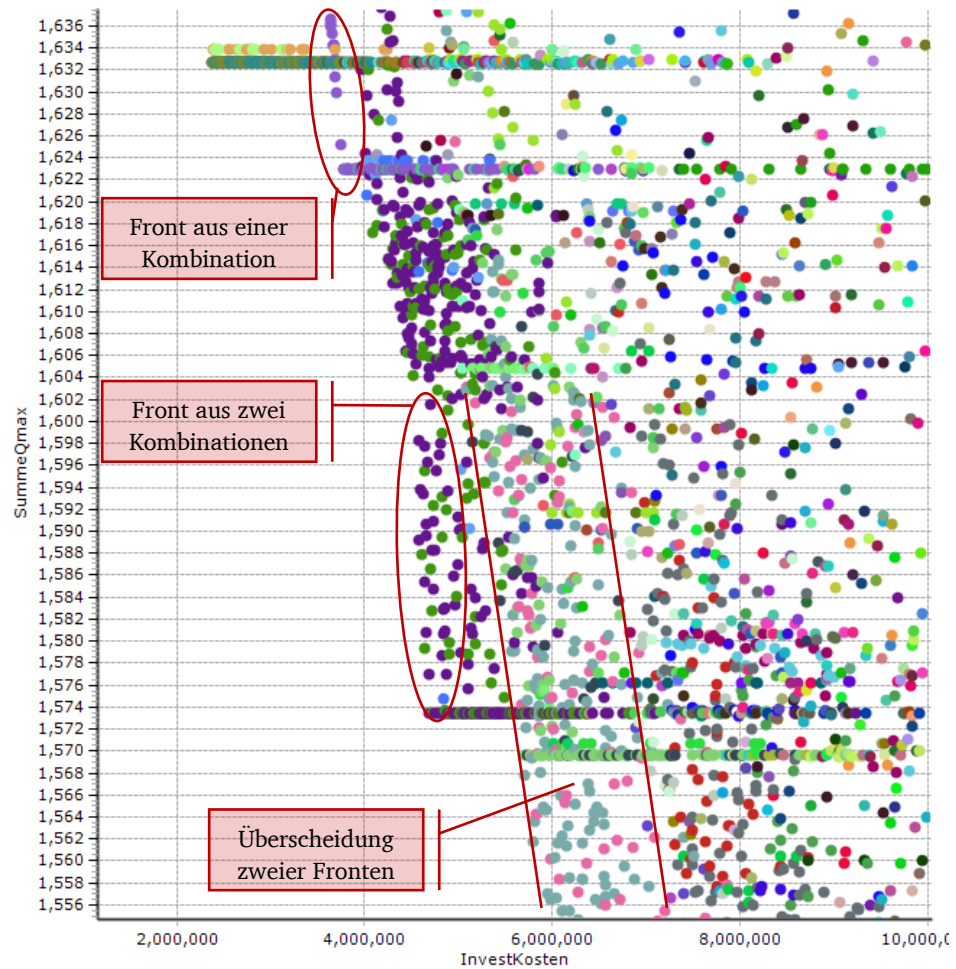


Abbildung 7-16: Ausschnitt aus der Memetisch Evolutionären Optimierung

Um die Paretofront genauer zu untersuchen, wurde in Abbildung 7-16 ein Ausschnitt aus eben dieser vergrößert. Die gleiche Farbe für die Lösungen stellt jeweils eine Kombination dar, sodass zu erkennen ist, ob eine Lösung durch Variation der nominellen oder durch Variation der reellen Parameter erzeugt wurde. Die Paretofront wird nicht immer aus einer Kombination von Maßnahmen gebildet. Im mittleren und unteren Bereich ist zu erkennen, dass die Paretofront aus zwei verschiedenen Kombinationen und der Variation ihrer reellen Parameter besteht. Bei detaillierter Betrachtung fällt auf, dass es sich um Kombinationen handelt, welche sich nur unwesentlich voneinander unterscheiden.

Zwischen den beiden parallelen Linien liegen Optimierungsergebnisse bestehend aus zwei verschiedenen Kombinationen. Es ist zu erkennen, dass sie ab einem Wert von $Z_1 > 1574 \text{ m}^3/\text{s}$ zwar nicht mehr Teil der Front ist, sich aber weiterhin stabil bleibt. Durch eine

anschließende Sensitivitätsanalyse der reellen Parameter könnte der Einflussbereich der reellen Parameter detailliert analysiert werden.

7.2.2.2 Optimierung der Retentionsvolumina und Abgaben bei drei Hochwasserereignissen

Für diese multikriterielle Optimierung mit drei Hochwasserereignissen wird eine Seriensimulation eingesetzt. Bei den Elementen der Berechnungsserie handelt es sich um die Einzelereignisse 1984/06, 1995/01 und 2000/10.

Da sie sich die Fragestellung auf das spezielle Problem Hochwasserabflusssimulation beschränkt, kann die in ATV (1985) definierte Seriensimulation verwendet werden. Allerdings ist zu beachten, dass die Unabhängigkeit der einzelnen Elemente gewährleistet ist. Die Trockenperioden müssen hinreichend groß gewählt werden, sodass die vorangehenden Ereignisse nicht den Füllungsgrad der Speicher beeinflussen. Unter Umständen kann die Forderung der Unabhängigkeit der Elemente dazu führen, dass die erforderlichen Simulationszeiten entscheidend verlängert werden. Die aktuelle Version des Optimierungswerkzeuges BlueM.Opt erlaubt es, Zeitreihen bereichsweise auszuwerten. Damit ist es möglich, die Abflussspitzen der Einzelereignisse mit nur einer Evaluierung zu ermitteln.

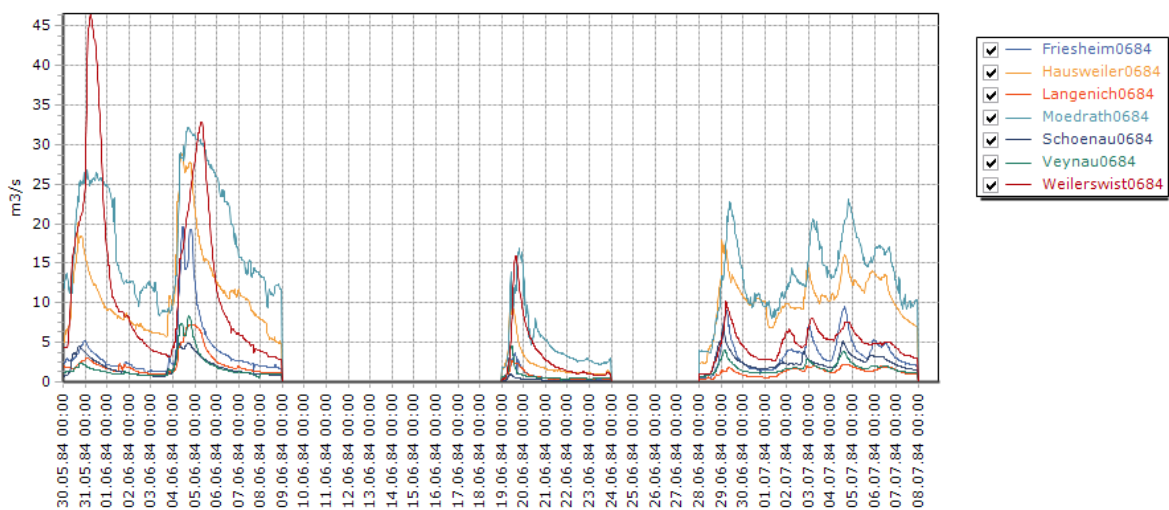


Abbildung 7-17: Zuflüsse der Seriensimulation mit 3 Ereignissen

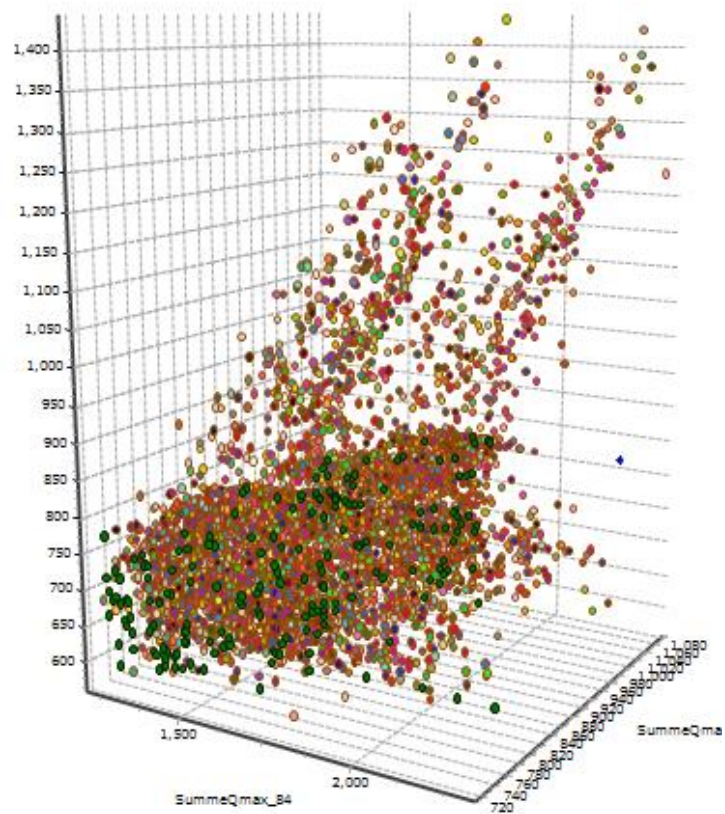


Abbildung 7-18: Darstellung des Optimierungsergebnisses mit drei von vier Zielfunktionen

Abbildung 7-18 zeigt den Optimierungsverlauf mit drei Zielfunktionen zur Reduzierung der Scheitel. Der Algorithmus hat sich in einen Bereich vorgearbeitet, der bei allen drei Ereignissen eine Reduktion des Hochwasserscheitels bewirkt. Die vierte Zielfunktion kann nicht mehr dargestellt werden.

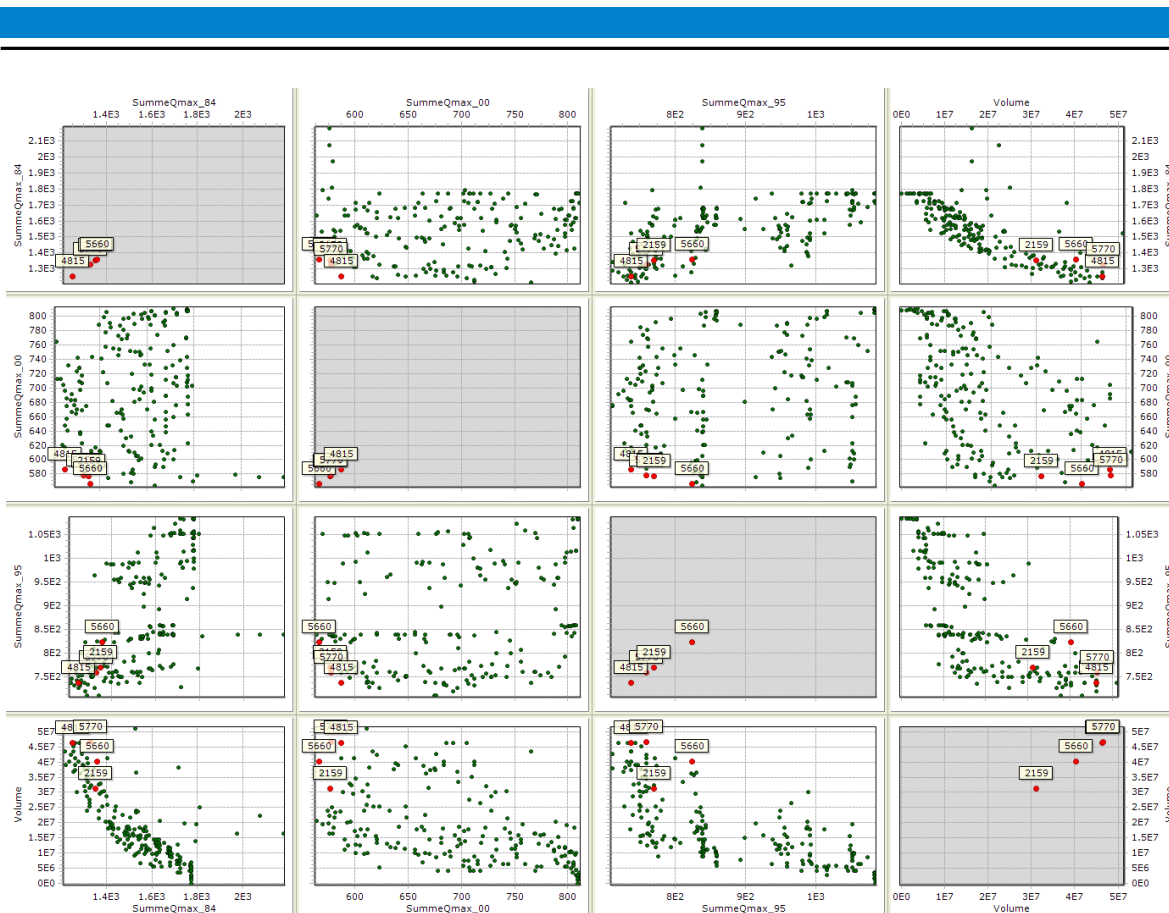


Abbildung 7-19: Scatter-Plot-Matrix des Optimierungsergebnisses

Um mehr als drei Zielfunktionen abbilden zu können, muss eine Scatter-Plot-Matrix (Abbildung 7-19) erstellt werden. In dieser Matrix werden die vier Dimensionen auf zwei-dimensionale Flächen projiziert, denn der Lösungsraum ist vierdimensional und keine Linie oder Fläche mehr, sondern ein Volumen. Interessant ist die rechte Spalte. Dort werden wie gewohnt die Scheitelreduktionen der Einzelereignisse und die Retentionsvolumina projiziert. In der obigen Abbildung sind vier „gute“ Lösungen markiert. Diese werden automatisch auch im drei-dimensionalen Diagramm angezeigt (Abbildung 7-20; es sind nur drei Ereignisse mit Nummern versehen; die Vierte ist rot markiert und liegt über der x – Achsenbeschriftung).

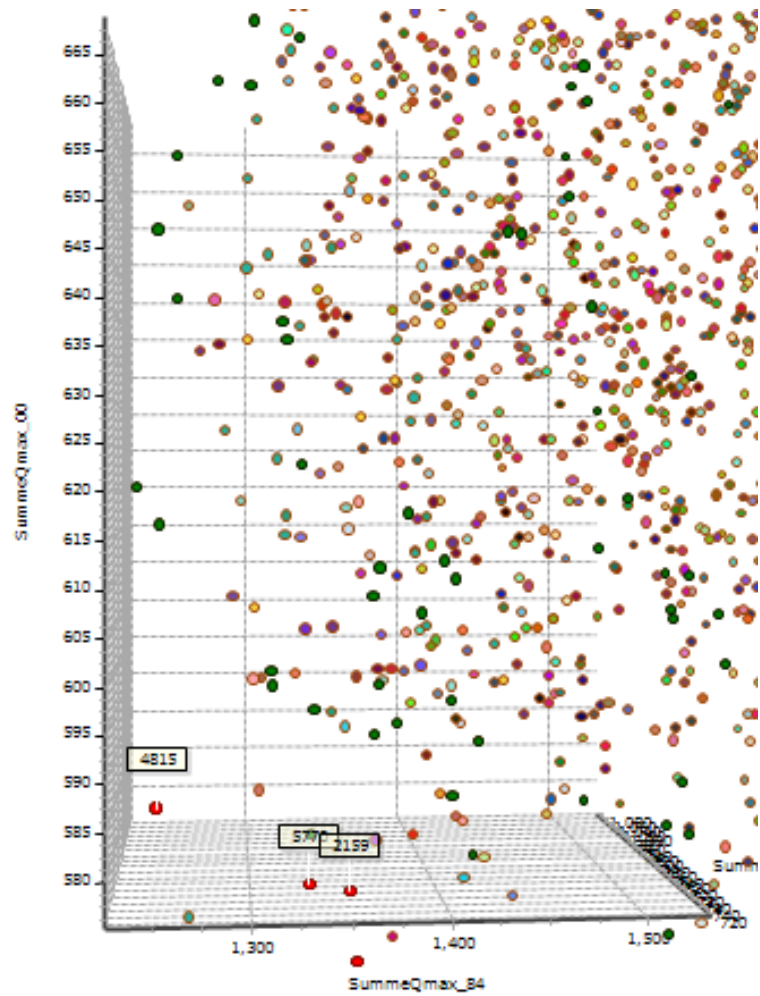


Abbildung 7-20: Vergrößerung des optimalen Bereiches

Die Ereignisse können bei Bedarf erneut simuliert werden um die Abflussganglinien (Abbildung 7-21) untersuchen zu können.

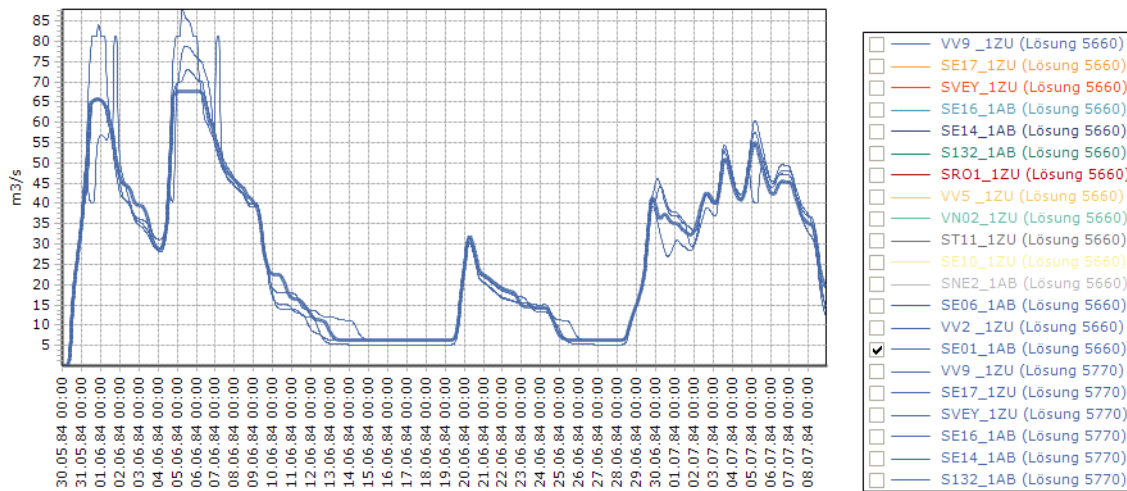


Abbildung 7-21: Abflussganglinien am Zielpegel der vier selektierten Lösungen

Man bedenke, dass alle Lösungen „beste“ Lösungen sind, da sie Teil der Paretofront sind. Die Ganglinie die mit der etwas stärkeren Strichstärke Abbildung 7-21 reduziert die Scheitel am besten, ist aber auch diejenige, die die meisten Hochwasserspeicher einsetzt. Auf den ersten Blick lassen sich die resultierenden Ganglinien nicht ohne weiteres nachvollziehen. Die Zielsetzung bei drei Ereignissen mit stark unterschiedlichen Jährlichkeiten ist bei konstanten Abgaben naturgemäß gegensätzlich. Die folgende Abbildung 7-22 zeigt die Hochwasserganglinien des „besten“ Ereignisses an vier verschiedenen Beobachtungspunkten.

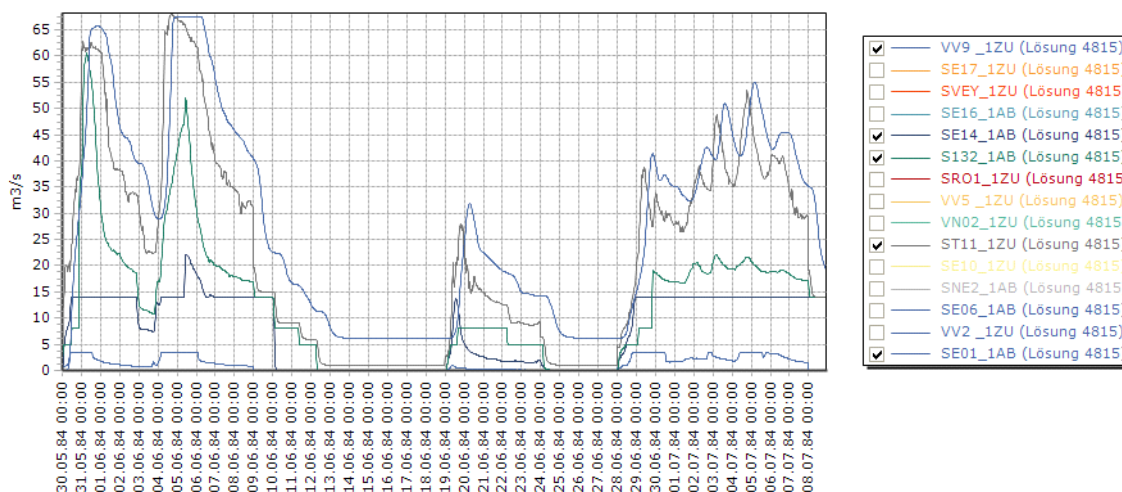


Abbildung 7-22: Abflussganglinien des "besten" Ereignisses an verschiedenen Beobachtungspunkten

Betrachtet man die Ganglien genauer, wird deutlich, warum der Algorithmus diese Lösung vorschlägt. Die Abgaben eines Speichers sind konstant. Daher setzt er die Hochwasserspeicher so ein, dass die Volumina und die Höhe der Abgaben für ein Ereignis wirksam werden und nimmt dabei in Kauf, dass die jeweilige Hochwasserentlastung der Speicher aktiv wird. Einen Hochwasserspeicher „überlaufen“ zu lassen, ist zwar häufig dem Hochwasserschutz zuträglich, aber nun dann, wenn gesicherte Prognosen über den Abfluss vorliegen. Läuft der Speicher zu früh voll, wird der Hochwasserscheitel überhaupt nicht reduziert. Diese Beispiel mit drei Hochwasserereignissen zeigt, dass ein evolutionärer Optimierungsalgorithmus interessante Lösungen aufzeigen kann. Eine Überprüfung im Nachgang bleibt aber unerlässlich.

7.3 Operatoren und Einstellungen

Detaillierte Vergleiche der Operatoren des nominell diskreten Algorithmus wurden in Kapitel 3.1.4 erprobt und in Kapitel 3.1.5 diskutiert. Daher werden hier im Wesentlichen einige Eigenschaften des memetisch evolutionären Algorithmus diskutiert. Insgesamt stehen 14 Reproduktions-, 8 Mutations-, 4 Selektionsoperatoren und 47 weitere Einstellungen (Mutationsrate, Größe des Speichers, Schrittweite, etc.) zur Verfügung. Eine Übersicht über die Einstellungen ist in der Anlage H zu finden. Da für die memetische Optimierung nahezu alle Einstellungen und Operatoren zur Verfügung stehen, kann das Thema „Vergleich der Operatoren und Einstellungen nicht abschließend behandelt werden. Gute Einstellungen zur Kontrolle der Algorithmen lässt sich als „*Optimierung der Optimierung*“ auffassen. Es folgen daher einige grundsätzliche Überlegungen dazu.

Die Optimierung der Optimierung

Der Wunsch nach der Optimierung der Optimierung liegt nahe. Wenn ein wie auch immer gearteter Optimierungsalgorithmus vorliegt der in der Lage ist, Modellparameter zu optimieren, warum soll dieser dann nicht auch gleich seine eigenen Parameter optimieren? Alle Operatoren und Einstellungen eines Algorithmus könnten mithilfe eines memetischen Optimierungsalgorithmus optimiert werden. Wenn man den dadurch entstehenden enormen Bedarf an Rechenzeit außer Acht lässt, ist das grundsätzlich möglich. Dabei werden alle Einstellungen und Operatoren in Form einer gemischt-ganzzahligen Repräsentation formuliert und durch einen übergeordneten Algorithmus optimiert. Dieses

Vorgehen ist dann möglich, wenn es nur um eine Zielsetzung geht, beispielsweise die Optimierungsgeschwindigkeit. Ein Metaqualitätswert könnte die Qualitätswertsteigerung pro Generation abbilden. Sicher wird eine Kombination von Einstellungen gefunden, die im Lösungsraum schneller voranschreitet als alle anderen. Unklar bleibt, ob bei der Optimierung der Geschwindigkeit auch das Optimum gefunden wurde oder die ermittelte Kombination nur schnell, aber nicht erfolgreich war. Daher müsste ein zweites Kriterium eingeführt werden, welches die Erfolgsquote abbildet. In diesem multikriteriellen Fall wäre aber eine automatisierte Optimierung nicht mehr möglich, da dann nicht mehr eine Lösung ermittelt wird, sondern eine Menge an Lösungen. Allgemein wird bei der Optimierung zunächst der Parameterraum modifiziert und im Lösungsraum abgebildet. Bei der Optimierung der Optimierung muss zusätzlich der Lösungsraum untersucht werden. Die Konsequenz ist eine kaskadenhafte Optimierungsprozedur (Rechenberg, 1994). Die entscheidende Frage dabei ist - so Rechenberg - mit welchen Werten die neue Optimierung gestartet wird, denn nach der ersten Optimierung befinden sich die Eltern bereits auf globalen Minima.

Allerdings hat sich diese kaskadenhafte Optimierung nicht durchgesetzt. Einer der bekanntesten Algorithmen, der NSGA-II (Non-Dominated Sorting Genetic Algorithm; (K. Deb u. a., 2000)) verfährt beim Thema Optimierung der Optimierung anders. Beispielsweise besitzt der in dieser Arbeit eingesetzte reellwertige Mutationsoperator einen Steuerparameter D_n , der die Schrittweite bei der Mutation definiert. Vor der Optimierung muss diese Schrittweite angegeben werden. Allerdings handelt es sich nur um die Startschrittweite. Im Laufe der Generationen wird diese Schrittweite bei der Reproduktion angepasst und der neue Wert den Nachfahren übergeben. Es handelt sich also um einen adaptiven Parameter, der ebenfalls der Evolution unterliegt und an die aktuellen Gegebenheiten des Lösungsraums angepasst wird. Das bedeutet, dass die Schrittweite während der Optimierung optimiert wird. Der NSGA-II Algorithmus verfährt auf diese Weise nicht nur bei der Mutation sondern auch bei der Reproduktion. Durch spezielle Operatoren wird der Selektionsdruck während der Optimierung angepasst. Ein wesentlicher Vorteil dieser Methode besteht auch darin, dass mit diesen Operatoren die Zahl der Stellschrauben reduziert werden kann. Beim NSGA-II muss vor der Optimierung lediglich die Populationsgröße (Anzahl der λ Nachfahren) bestimmt werden. Die Anzahl der Eltern μ bestimmt der Algorithmus selbst. Die Zahl der Stellschrauben muss vor allem reduziert werden um dem

reinen Anwender den Zugang zu Algorithmen dieser Klasse zu erleichtern. Daher muss die Optimierung der Optimierung Teil des „normalen“ Optimierungsprozesses sein.

7.3.1 Anpassung der Schrittweite bei der memetischen Optimierung

Wie bereits erwähnt ist für eine erfolgreiche Memetische Optimierung die nominelle Mutationsrate sehr sensitiv. Der nominellen Mutationsrate entspricht beim reellen Anteil die Schrittweite D_n . Diese ist adaptiv, das heißt, sie ist Teil des Individuums und wird in jeder Generation vererbt und mutiert. Sind in der aktuellen Generation Individuen mit großer Schrittweite ($> 0,1$) erfolgreich, weil sie sich dem Optimum schneller nähern können, so wird diese für die Selektion ausgewählt. Ist es erforderlich mit kleinen Schrittweiten ($< 0,1$) an das Optimum anzunähern werden Individuen mit kleinem D_n bevorzugt.

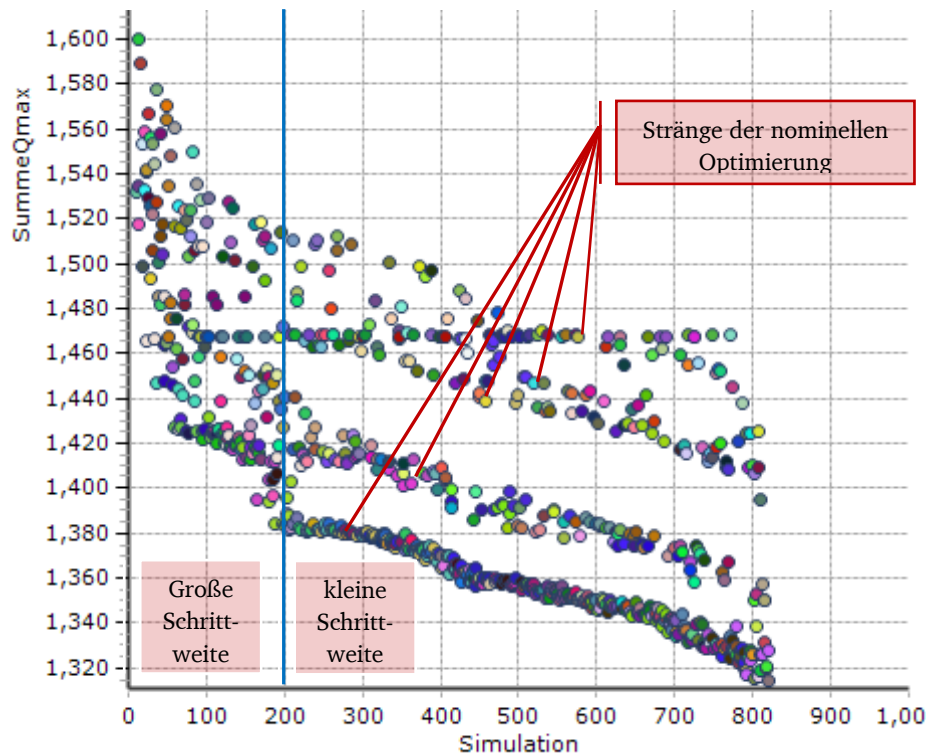


Abbildung 7-23: Optimierungsverlauf bei starker Reduzierung der reellen Schrittweite

Abbildung 7-23 zeigt wie sich der memetische Algorithmus verhält, wenn die Schrittweite durch die Selektion abrupt verringert wird. Bei Simulation 200 wurde die Mutations-schrittweite D_n wahrscheinlich auf Grund eines lokalen Minima stark reduziert. In der Folge werden die reellen Variablen nur noch geringfügig modifiziert. Dadurch werden die einzelnen Stränge der nominellen Optimierung sichtbar. Im Bereich mit großer Schritt-

weite schreitet die Optimierung zwar weiter voran, allerdings erfolgt dies mit einer wesentlich kleineren Geschwindigkeit.

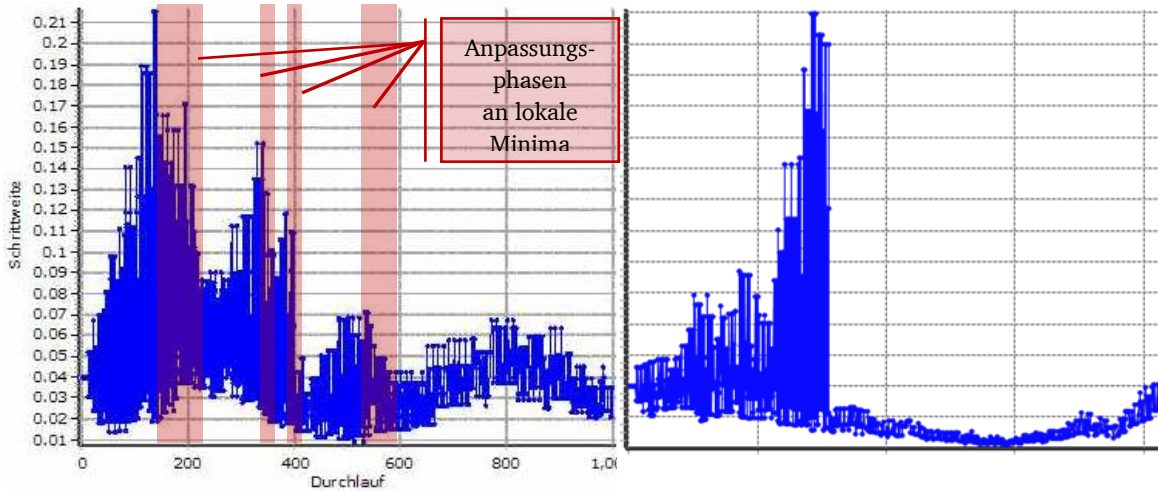


Abbildung 7-24: Kontinuierlicher D_n Verlauf

Abbildung 7-25: D_n Verlauf mit abrupter Änderung

Abbildung 7-24 und Abbildung 7-25 vergleichen die Verläufe der Schrittweiten. In der linken Abbildung wurde die Größe und Spannweite der Schrittweite im Optimierungsverlauf kontinuierlich angepasst. An den Stellen wo sich die Spannweite verringert existieren lokale Minima. Im rechten Verlauf wurde die Schrittweite beim ersten lokalen Minima zufällig stark reduziert (Dieses Ergebnis konnte in weiteren Versuchen nicht reproduziert werden). Gegen Ende der Optimierung werden die Größe und die Spannweite der Schrittweite wieder leicht erhöht, sie kann sich aber nicht mehr vollständig erholen.

7.3.2 Steuerung der Selektion \mathcal{E}_{mem}

Im Kapitel 3.2.4.3 „Parallele Optimierung mit selektierten reellen Parametern“ wurde eine zusätzliche Einstellung für die memetische Selektion \mathcal{E}_{mem} eingeführt. Der Einfluss der Settings wurde in ca. 100 Optimierungsläufen verglichen.:

- "Two-Locations-Up"
- "One-Location-Up"
- "This-Location"
- "One-Location-Down"
- "Two-Locations-Down"

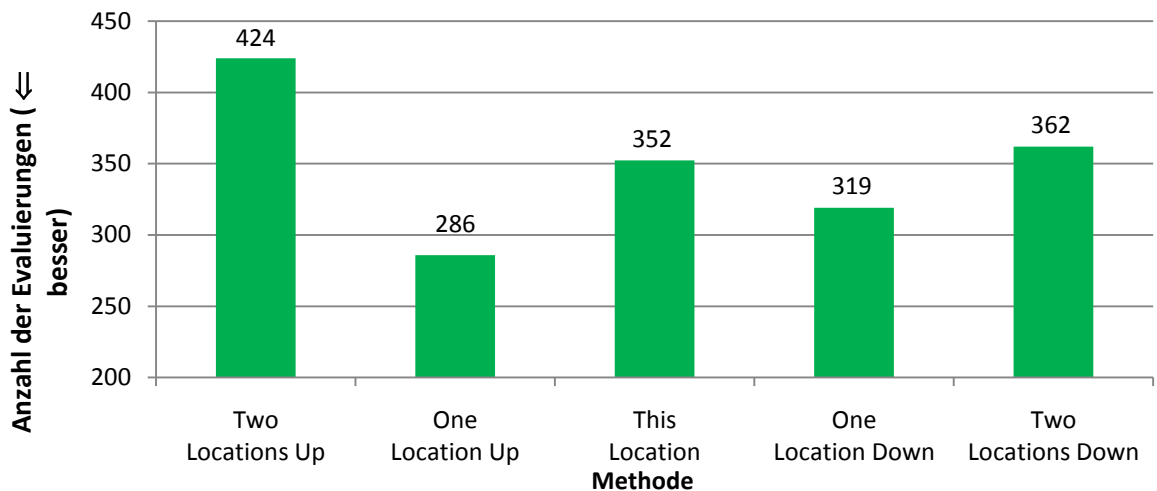


Abbildung 7-26: Vergleich der Settings für den \mathcal{E}_{mem} Operator

Abbildung 7-26 zeigt ein uneinheitliches Ergebnis. Zwar wird deutlich, dass dieser Steuerparameter einen Einfluss auf die Optimierungsgeschwindigkeit hat, es ist aber kein Trend erkennbar. Das ist dadurch bedingt, dass die Repräsentation keine Information darüber enthält welche Position die nachfolgende und welche Position die vorhergehende ist. Dadurch werden bei Standorten in Seitenarmen falsche Beziehungen angenommen. Am effektivsten ist es bei der memetischen Selektion \mathcal{E}_{mem} den flussauf liegenden Hochwasserspeicher zu berücksichtigen. Um das Ergebnis aber abschließend beurteilen zu können wäre es erforderlich, eine Information über Vorgänger und Nachfolger im Individuum zu hinterlegen. Dann könnten ggf. auch entsprechende Reproduktions- und Mutationsoperatoren definiert werden.

7.3.3 Lokaler und globaler Lösungsraum

In Kapitel 3.2 wurde der memetische Algorithmus beschrieben und unter anderem auf die nominell globale Optimierung eingegangen, welche lokal von der reell globalen Suche unterstützt wird.

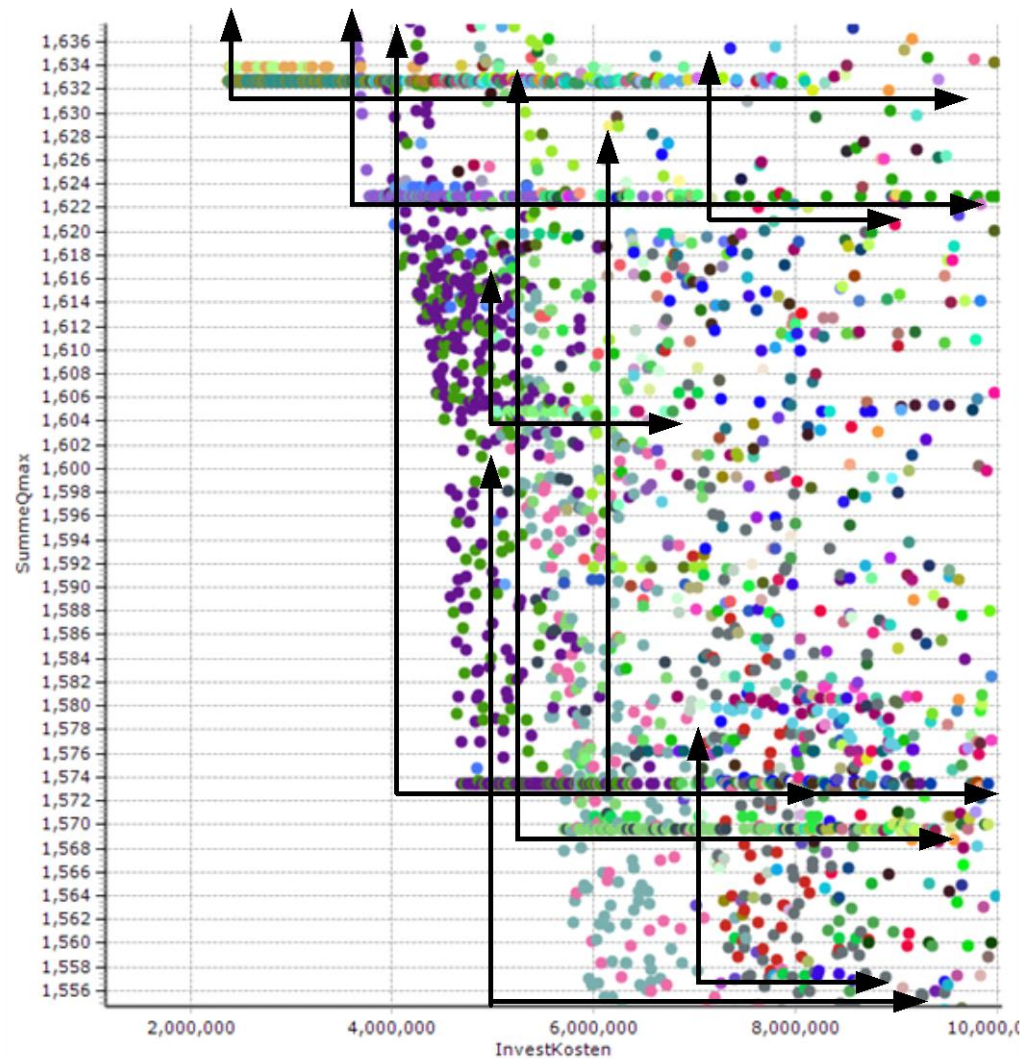


Abbildung 7-27: Lokale reelle Lösungsräume

Um diese Eigenschaft des Lösungsraumes zu verdeutlichen, wurden in Abbildung 7-27 die lokalen Lösungsräume im Gesamtlösungsraum des Anwendungsbeispiels der Erft kenntlich gemacht. Die schwarzen Pfeile stellen hierbei die Achsen der Sublösungsräume dar. Die gewählte Architektur des memetischen Algorithmus ist also dafür geeignet mit derartig inhomogenen Lösungsräumen umzugehen.

8 Fazit und Ausblick

In dieser Arbeit wurde ein Memetischer Evolutionärer Algorithmus zur Optimierung von Speichersystemen entwickelt. Dieser ermöglicht es erstmalig Hochwasserspeicher in ihrer Lage, Größe und Steuerung zu optimieren.

Zu diesem Zweck wurde zunächst ein nominell diskreter evolutionsstrategischer Optimierungsalgorithmus entwickelt, dessen Funktionsfähigkeit anhand des mathematischen Rundreiseproblems bewiesen wurde. Aufbauend auf einer Analyse der Problemdomäne Hochwasserschutz wurde der entwickelte nominell diskrete Algorithmus zur Optimierung von Hochwasserspeichersystemen erweitert. Mit geeigneten Reproduktions- und Mutationsoperatoren war es möglich, die betrachtete Problemstellung im Einzugsgebiet der Erft monokriteriell als auch multikriteriell zu lösen. Die Hochwasserspeicher konnten nominell in der Lage sehr schnell und zuverlässig optimiert werden. Sowohl bei der monokriteriellen, als auch bei der multikriteriellen Optimierung konnte der Algorithmus seine Leistungsfähigkeit unter Beweis stellen.

In einem weiteren Schritt wurde ein neuer Memetischer Evolutionärer Algorithmus entwickelt, der zur Optimierung nominal diskrete Operatoren als auch reellwertige Operatoren einsetzt. Für die memetische Optimierung wurde ein neuer Populationspeicher P_{mem} , und ein neuer Selektionsoperator E_{mem} eingeführt, um die für die Hochwasserspeicheroptimierung erforderlichen beiden Variablentypen optimieren zu können. Der entwickelte Memetische Evolutionäre Algorithmus erlaubt, es die Lage, Volumina und Abgaben der Hochwasserspeicher schnell und zuverlässig zu optimieren. Allerdings muss darauf geachtet werden, dass die Anpassung der reellwertigen Parameter nicht von den nominal diskreten Optimierungsprozessen überschattet wird. Der für die Kombination verantwortliche Genotyp aus nominal diskreten Parametern darf in jeder Generation nicht zu stark mutiert werden. Anwendbarkeit und Funktionsfähigkeit wurden anhand des Praxisbeispiels Erft bewiesen. Die hier entwickelte memetische Methode konnte Ihre Leistungsfähigkeit bei einer multikriteriellen Fragestellung mit vier Zielfunktionen eindrucksvoll unter Beweis stellen.

Im gewählten Anwendungsbeispiel konnte gezeigt werden, dass auf Basis des verwendeten Simulationsmodells im gewählten Anwendungsbeispiel effektivere Lösungen existieren als

die bestehenden Hochwasserschutzmaßnahmen. Insbesondere die multikriterielle Optimierung zeigt, dass es Kombinationen aus Speichern gibt, die bei gleichen Retentionsvolumina einen besseren Hochwasserschutz oder den gleichen Schutzgrad bei weniger Volumina ermöglichen.

Durch den Einsatz von Investitionskostenfunktionen konnten mit dem nominell diskreten, als auch mit dem memetischen Algorithmus Lösungsmengen von Hochwasserspeicherkombinationen in Form von Paretofronten ermittelt werden, die für die Planung von Maßnahmen eine wertvolle Informationsquelle sind. Durch das multikriterielle Verfahren kann ein Entscheider frei nach seiner Präferenz zwischen einem größeren Hochwasserschutz zu einem höheren Preis oder dem besten Hochwasserschutzgrad bei einem limitierten Budget wählen. Diese Wahl kann, eine fachgerechte Simulation und Optimierung vorausgesetzt, in der Gewissheit erfolgen, dass die gewählte Maßnahmenkombination die Hochwasserscheitel stets effektiv reduzieren kann.

Des Weiteren wurde im Einzugsgebiet der Erft nach Lösungen gesucht, die in der Lage sind, einen guten Hochwasserschutzgrad bei Hochwasserereignissen mit höchst unterschiedlichen Wiederkehrintervallen zu ermöglichen. Obwohl dies bei einer Steuerung der Speicher mit konstanten Abgaben nur bedingt möglich ist, konnte gezeigt werden, dass der Algorithmus die Interaktionen der Maßnahmen nicht nur berücksichtigt, sondern diese gezielt zur Reduzierung der Scheitel aller Ereignisse gezielt einsetzt. Optimierungsergebnisse sind zwar sehr informativ aber unter Umständen auch tückisch und müssen stets im Nachgang detailliert geprüft werden.

Zur Reduzierung der Optimierungszeit wurde das Optimierungsverfahren parallelisiert. Die gewählte Parallelisierungsmethode erlaubt es, die Fähigkeiten aktueller Mehrkernprozessoren gänzlich auszuschöpfen. Es wurde experimentell nachgewiesen, dass der Optimierungsprozess unter Verwendung eines Prozessors mit zwei Kernen um den Faktor 2,3 beschleunigt wird. Unter der Voraussetzung, dass entsprechend viele Prozessoren zur Verfügung stehen, wurde rechnerisch nachgewiesen, dass die maximale Beschleunigung der Optimierung bei dem Faktor 20 liegt. In Verbindung mit dem experimentellen Ergebnis scheint eine Beschleunigung der Optimierung durch die gewählte Parallelisierung um einen Faktor, der größer als 20 ist, als realistisch.

Bei der memetischen Optimierung wurden Operatoren für reellwertige Repräsentationen als auch Operatoren für nominell diskrete Repräsentationen verwendet. Manche Operatoren besitzen bereits eigene Steuerparameter. Zusätzlich sind für die memetische Optimierung weitere Steuerparameter erforderlich. Damit steigt auch die Zahl der „Stellschrauben“, die es vor jeder Optimierung anzupassen gilt, um gute Ergebnisse zu erzielen. Beispielsweise geht eine Erhöhung der Optimierungsgeschwindigkeit mit der Gefahr einher, das globale Optimum nicht zu finden. Daher bedarf es einiger Erfahrung im Umgang mit und in der Entwicklung von Algorithmen dieser Art.

Der Informationsgewinn liegt nicht nur in den resultierenden Optima. Bei selbstlernenden Algorithmen dieser Art besteht der Erkenntnisgewinn vor allem darin, nachzuvollziehen, warum diese sich für einen Gradientenweg entscheiden. Daraus geht zum einen hervor, ob die gewählten Zielsetzungen sinnvoll sind und das Modell hinreichend ist, um die Problemstellung zu lösen. Zum anderen erhält man ein umfassendes Bild des Lösungsraumes. In manchen Fällen kann aufgrund der Charakteristik des Lösungsraumes darauf geschlossen werden, dass einer der Modellparameter starken Einfluss auf die Simulationsergebnisse haben musste. Ebenso zeigt das multikriterielle Ergebnis der Optimierung mit drei Hochwasserereignissen, dass Hochwasserereignisse unterschiedlicher Größenordnungen nur bedingt mittels konstanter Speicherabgaben kontrollierbar sind. Das Problem wurde vom Algorithmus gelöst, in dem er zum einen mehrere Speicher einsetzte und deren Abgaben und Volumina gezielt auf eines der drei Ereignisse ausrichtete.

Ein wesentlicher Vorteil der evolutionären Algorithmen ist, dass das verwendete Modell nicht angepasst und in seiner Komplexität reduziert werden muss. Im Vergleich zur dynamischen Programmierung muss der Entscheidungsraum nicht diskretisiert werden und keine Trajektorien zur Initiierung ermittelt werden. Bewährte Modelle können an das Optimierungssystem angebunden werden. Diese Flexibilität ist ein wesentlicher Vorteil bei der Anwendung in der Praxis. Das bedeutet, es können weiterhin die Modelle verwendet werden, die dem praxisorientierten Anwender bekannt sind und deren Zuverlässigkeit mehrfach bestätigt wurde.

Generell hat die Anwendung von Optimierungstechniken bei in der Planung einen wesentlichen Vorteil. Man ist weniger stark auf die Entwicklung einzelner Varianten angewiesen und kann damit besser auf im Planungsverlauf erforderliche Anpassungen reagieren. Dis-

kussionen sind dann nicht mehr auf das Pro und Kontra von Planungsvarianten beschränkt und es ist sichergestellt, dass unter den gegebenen Rahmenbedingungen stets die besten Lösungen zur Auswahl stehen.

Weitere Anwendungsbereiche

Zur Berechnung von Hochwasserschäden können Schadenfunktionen eingesetzt werden. Das vorhandene Optimierungssystem erlaubt es, beliebige funktionale Zusammenhänge als Zielfunktionen einzusetzen, wie sie beispielsweise zur Berechnung der Investitionskosten eingesetzt wurde. In Verbindung mit den Investitionskosten kann ein Kosten-Nutzen-Wert berechnet werden. Dies hätte auch den Vorteil, dass die hier angestellten Betrachtungen mit den Zielen der Reduktion des Hochwasserscheitels und der Minimierung der Kosten auf eine Zielsetzung, das Nutzen-Kosten-Optimum, reduziert werden könnte. Auf diese Weise würde zum einen der Aufwand für die Optimierung verringert, zum anderen würde die Optimierung zu einer einzelnen Lösung führen und nicht zu einer Lösungsmenge in Form der Paretofront.

Mit der in dieser Arbeit entwickelten Methode ist es auch möglich, die Optimierung der Ausbaureihenfolge vorzunehmen. Für Einzugsgebiete werden häufig übergeordnete Planungen angestrengt, die den Ausbau des Hochwasserschutzes über mehrere Dekaden vorsehen. Diese Planungen sehen meist nicht nur eine Maßnahme vor, sondern mehrere kleinere oder größere Maßnahmen innerhalb eines Hochwasserschutzkonzeptes. Bei dieser Planung eröffnet sich die Frage, welche Maßnahmen vorrangig realisiert werden sollen, da durch diese von Beginn an ein möglichst großer Hochwasserschutzgrad gewährleistet ist. Um diese Frage zu beantworten, kann mit dem hier entwickelten nominell diskreten Algorithmus zunächst eine gute Kombination an Maßnahmen ermittelt werden. Damit stehen die übergeordnete Zielsetzung und das anvisierte Schutzziel fest. Im Anschluss kann durch $\sum_{i=2}^n i$ mit $n = \text{Anzahl der Standorte}$ Simulationen die Reihenfolge bestimmt werden, in der die Maßnahmen am besten realisiert werden sollten, um den größten Nutzen zum Zwecke des Hochwasserschutzes zu erzielen. Sollen zusätzlich die Abgaben und Volumina optimiert werden, ist es mit dem memetischen Algorithmus möglich, eine gute Lösung zu bestimmen. Die Ausbaureihenfolge kann im Anschluss durch $\sum_{i=2}^n i$ rein reellwertige Optimierungen bestimmt werden.

Ausblick

Die Einsatzmöglichkeiten der in dieser Arbeit entwickelten Methoden sind nicht auf den Themenbereich Hochwasserschutz begrenzt. Der Algorithmus kann auf beliebige gemischt ganzzahlige Problemstellungen wie zum Beispiel zur Planung von Kanalnetzen oder Gasnetzen verwendet werden.

Mit der aktuellen Version des Optimierungswerkzeuges können reelle als auch nominal diskrete Variablen optimiert werden. Der Algorithmus könnte künftig mit Operatoren ausgestattet werden, die mit ordinal diskreten Variablen (beispielsweise für Rohrdurchmesser die nur in bestimmten Normgrößen hergestellt werden) arbeiten können. Da diese ordinal diskreten Operatoren den reellen Operatoren sehr ähnlich sind, ist es nicht erforderlich neue Prozeduren zu entwickeln. Erforderlich wäre es allerdings die ordinal diskrete Mutation mit der reellen Mutation mithilfe eines Faktors zu synchronisieren, da durch die diskreten Abstände, wie bei der nominal diskreten Optimierung, Sprünge im Lösungsraum entstehen. Zusätzlich ausgestattet mit diesen ordinal diskreten Operatoren, wäre der Optimierungsalgorithmus geeignet um die drei häufigsten Variablentypen, kontinuierlich, ordinal diskret und nominal diskret zu optimieren.

Bei den Optimierungsläufen dieser Arbeit war in einigen Fällen eine erhebliche Streuung der Nachfahren kurz vor dem globalen Optimum nicht zu verhindern. Der Grund sind die diskreten Parameter, die zu Sprüngen im Lösungsraum führen. Es könnte untersucht werden, ob es möglich ist, diese Streuung zu verhindern indem Maßnahmen, die seit einigen Generationen nicht mehr verwendet wurden während des Prozesses von der Optimierung ausgeschlossen werden. Dies könnte ebenfalls für reelle Parameter untersucht werden, indem man mit einem D_n Vektor arbeitet und beobachtet, wie sich die Schrittweite der einzelnen Parameter verhält. Ein Wert für den D_n der sich über längere Zeit weit über eins bewegt, deutet darauf hin, dass dieser Modellparameter kaum Sensitivität besitzt. Diese Parameter könnten ebenfalls von der Optimierung ausgeschlossen werden. Auf diese Weise ließe sich der Lösungsraum sukzessive verkleinern, mit der Konsequenz, dass die Optimierungsgeschwindigkeit zunimmt, aber die Wahrscheinlichkeit das globale Optimum zu finden sinkt.

Bezüglich der nominell diskreten Mutationsrate kann empfohlen werden, dass es künftig erforderlich ist, die Rate in Promille von der Größe des Genotyps und nicht wie bisher in Prozent einzustellen. Diese Stellschraube hat sich als extrem sensitiv erwiesen.

Andere Problemstellungen machen es unter Umständen erforderlich mit hydraulischen Modellen zu arbeiten. Bekanntermaßen sind hydraulische Modelle sehr rechenzeitintensiv. Auch um die Rechenzeiten in einem für die Optimierung verträglichen Rahmen zu halten, war es in dieser Arbeit erforderlich, ein hydrologisches Modell zur Abflussberechnung einzusetzen. Durch die in dieser Arbeit vorgenommene Parallelisierung der Simulationen steht nun eine Methode zur Verfügung, die erheblich dazu beiträgt die Rechenzeiten für die Optimierung zu verkürzen. Falls ein hydraulisches Modell dafür geeignet ist, mehrfach parallel ausgeführt zu werden, sollte es unter der Berücksichtigung der Performanz des Modelldatensatzes auch möglich sein, mit der hier vorgestellten Methode Problemstellungen zu lösen, die eine hydraulische Simulation erfordern.

Abschließend kann gesagt werden, dass die Anwendung Memetischer Evolutionärer Algorithmen im ingenieurtechnischen Bereich zu empfehlen ist. Die Algorithmen arbeiten hinreichend schnell und robust und sind durch ihre Flexibilität dazu geeignet, ein weites Feld von Problemstellungen zu lösen. Bei memetischen Algorithmen wird auch die in die biologische Evolution eingebettete kulturelle Evolution berücksichtigt. Erst durch dieses Verfahren wurde es möglich, die komplexen ingenieurtechnischen Fragestellungen, die in dieser Arbeit gestellt wurden, zu beantworten.

Literaturverzeichnis

- Alex, W. und Ganten, P. H. (2007) *Debian GNU/Linux: Grundlagen, Einrichtung und Betrieb*, Springer.
- AMD, Inc. (2009) AMD Developer Central - AMD CodeAnalyst. AMD Developer Central - AMD CodeAnalyst. [online] <http://developer.amd.com/CPU/CODEANALYST/Pages/default.aspx> (Zugegriffen April 24, 2009).
- Amdahl, G. M. (1967) "Validity of the single processor approach to achieving large scale computing capabilities" in Proceedings of the April 18-20, 1967, spring joint computer conference. Atlantic City, New Jersey, ACM, 483-485. [online] <http://portal.acm.org/citation.cfm?id=1465560> (Zugegriffen April 25, 2009).
- Amt für Wasserwirtschaft (Ost-Berlin) (1966) *Gewässerkunde - quantitativ Fachausdrücke und Begriffserklärungen*,
- Arnell, N. W. und Reynard, N. S. (1996) The effects of climate change due to global warming on river flows in Great Britain. *Journal of Hydrology*, 183(3-4), 397-424.
- ATV (1985) Langzeitsimulation - Begriffsbestimmung unter Berücksichtigung besonderer Anwendungsgebiete -, Arbeitsbericht der ATV-Arbeitsgruppe 1.2.6. Korrespondenz Abwasser, 32(4), 349 - 350.
- Bach, M., Froehlich, F., Heusch, S., Hübner, C., Muschalla, D., Reussner, F., u. a. (2009) "BlueM – a free software package for integrated river basin management" in N. Fohrer, B. Schmalz, G. Hörmann, und K. Bieger (hrsg.), *Hydrologische Systeme im Wandel, Beiträge zum Tag der Hydrologie 2009*. Theodor-Heuss-Allee 17, D-53773 Hennef, Fachgemeinschaft Hydrologisch Wissenschaften in der DWA, 109-115. [online] <http://www.fghw.de/>.
- Bach, M., Fröhlich, F., Huebner, C., Muschalla, D., Heusch, S., und Reußner, F. (2008) BlueM Beschreibung. unveröffentlicht.
- Balena, F. (2004) *Programming Microsoft® Visual Basic® .NET Version 2003*, Microsoft Press.
- Banzhaf, W. (1990) The "molecular" traveling salesman. *Biological Cybernetics*, 64(1), 7-14.
- Barricelli, N. A. (1954) Esempi numerici di processi di evoluzione. *Methodos*, 45–68.
- Barricelli, N. A. (1957) Symbiogenetic evolution processes realized by artificial methods. *Methodos*, 143–182.
- Bauke, H. und Mertens, S. (2006) *Cluster computing*, Springer.
- Baumgartner, V. (1980) *Optimale Steuerung von Speichersystemen im Hochwasserfall*, München, Bayer. Landesamt für Wasserwirtschaft.
- Bengel, G., Baun, C., Kunze, M., und Stucky, K. (2008) *Masterkurs Parallele und Verteilte Systeme: Grundlagen und Programmierung von Multicoreprozessoren, Multiprozessoren, Cluster und Grid*, Vieweg+Teubner.

- Beyer, H. G. (1992) "Some aspects of the evolution strategy for solving TSP-like optimization problems appearing at the design studies of a 0.5 TeV e+e-linear collider" in Manner and Manderick., 351-360.
- Beyer, H. und Deb, K. (2000) "On the Desired Behaviors of Self-Adaptive Evolutionary Algorithms" in Parallel Problem Solving from Nature PPSN VI., 59-68. [online]
http://dx.doi.org/10.1007/3-540-45356-3_6 (Zugegriffen September 2, 2009).
- Bogardi, J. (1979) *Optimale Ausbaureihenfolgen für Systeme von Hochwasserrückhaltebecken*, Karlsruhe, Universität Karlsruhe/ Institut Wasserbau III.
- Brass, V. V. C. (2006) *Betriebsoptimierung von Talsperrensystemen mittels Stochastisch Dynamischer Programmierung (SDP) unter Berücksichtigung veränderlicher Ziele und Randbedingungen*, Bochum, Univ.
- Brockhaus (2009) "FORTRAN" in Enzyklopädie in 30 Bänden. Mannheim, Bibliographisches Institut & F.A. Brockhaus AG.
- Bronstert, A., Fritsch, U., und Katzenmaier, D. (2001) Quantifizierung des Einflusses der Landnutzung und-bedeckung auf den Hochwasserabfluss in Flussgebieten unter Berücksichtigung der aktuellen Bodenbearbeitung, des Zustandes der Vegetationsdecke und möglicher Klimaänderungen. Report for the German Federal Environmental Agency, Berlin, 216.
- Bultot, F. (1988) Repercussions of a CO2 doubling on the water cycle and on the water balance — A case study for Belgium. *Journal of Hydrology*, 99, 319-347.
- Bultot, F., Gellens, D., Spreafico, M., und Schadler, B. (1992) Repercussions of a CO 2 Doubling on the Water Balance—A Case Study in Switzerland. *Journal of Hydrology JHYDA* 7,, 137(1/4).
- Bundesamt für Zivilschutz, Bern (1995) *Katastrophen und Notlagen in der Schweiz, eine vergleichende Übersicht (KATANOS)*.
- Cavalli-Sforza, L. L. (1971) Similarities and dissimilarities of sociocultural and biological evolution. *Mathematics in the archaeological and historical sciences*, 535–541.
- Chandra, R., Dagum, L., und Kohr, D. (2000) *Parallel Programming in OpenMP*, Morgan Kaufmann.
- Chuan Shi, Ming Chen, und Zhongzhi Shi (2005) "A Fast Nondominated Sorting Algorithm" in *Neural Networks and Brain, 2005. ICNN&B '05. International Conference on.*, 1605-1610.
- Davis, L. (1985) "Applying adaptive algorithms to epistatic domains" in *Proceedings of the International Joint Conference on Artificial Intelligence*. Los Angeles, California: IEEE Computer Society Press, 161–163.
- Dawkins, R. (1976) *The selfish gene*. Paladin, London.
- De Jong, K. A. (1975) Analysis of the behavior of a class of genetic adaptive systems. [online]
<http://deepblue.lib.umich.edu/handle/2027.42/4507> (Zugegriffen Juli 24, 2009).
- Deb, K., Agrawal, S., Pratab, A., und Meyarivan, T. (2000) *A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II, KanGAL Report 200001*, Kanpur, Indian Institute of Technology.

-
- Deltares (2009) Delft Hydraulics Software -Sobek. Delft Hydraulics Software.
- Dietrich, J. (2006) Entwicklung Einer Methodik Zur Systemanalytischen Unterstützung Adaptierbarer Entscheidungsprozesse Bei Der Integrierten Flussgebietsbewirtschaftung.
- Dochain, D. und Vanrolleghem, P. (2001) *Dynamical modelling and estimation in wastewater treatment processes*, IWA Publishing.
- Doyle, J. R., Heaney, J. P., Huber, W. C., und Hasan, S. M. (1976) Efficient Storage of Urban Storm Water Runoff.
- Du, J., Cai, Z., und Chen, Y. (2007) "A Sorting Based Algorithm for Finding a Non-dominated Set in Multi-objective Optimization" in International Conference on Natural Computation. Los Alamitos, CA, USA, IEEE Computer Society, 436-440.
- Dyck, S. (1980a) *Angewandte Hydrologie. Teil 1: Berechnung und Regelung des Durchflusses der Flüsse*, Berlin, Muenchen: Verlag von Wilhelm Ernst und Sohn.
- Dyck, S. (1980b) *Angewandte Hydrologie. Teil 2: Der Wasserhaushalt der Flußgebiete. 2. überarbeitete Aufl*, Berlin, VEB Verlag für Bauwesen. Berlin, München: Verlag W. Ernst & Sohn.
- Dyck, S. und Peschke, G. (1995) *Grundlagen der Hydrologie*, Berlin, Verlag für Bauwesen.
- Dyson, G. (2009) Retrospective - Barricelli's Universe. Make, 8. [online] <http://www.make-digital.com/make/vol08/?pg=198&u1=texterity> (Zugegriffen Dezember 19, 2009).
- EM-DAT (2010) EMDAT - The International Emergency Disasters Database. [online] <http://www.emdat.be/> (Zugegriffen Oktober 11, 2009).
- Erftverband (2009) Hochwasserrückhaltebecken (HRB) Garsdorf. [online] http://www.erftverband.de/projekte/rueck_becken/garsdorf/garsdorf.shtml (Zugegriffen November 20, 2009).
- Erftverband (2007) Hochwasserrückhaltebecken Mödrath. [online] http://www.erftverband.de/projekte/rueck_becken/moedrath/moedrath.shtml (Zugegriffen November 20, 2009).
- Erftverband (2006) Hochwasserrückhaltebecken Niederberg. [online] http://www.erftverband.de/projekte/rueck_becken/niederberg/niederberg.shtml (Zugegriffen November 20, 2009).
- Erftverband (2008) Pressemitteilung Erftverband. [online] http://www.erftverband.de/presse/10_07_08-4/presse.shtml (Zugegriffen November 20, 2009).
- Feldmann, L. (2009) Optimierung des Mitteleinsatzes für Hochwasserschutzmaßnahmen an Fließgewässern im urbanen Raum / Dissertation. [online] <http://opus.kobv.de/btu/volltexte/2009/1137/> (Zugegriffen November 30, 2009).
- Fogel, L. J. (1966) *Author Artificial intelligence through simulated evolution*. New York: John Wiley & Sons, 1966.

-
- Fox, M. S. und McMahon, M. B. (1987) "Genetic Operators for Sequencing Problems" in Foundations of Genetic Algorithms: First Workshop on the Foundations of Genetic Algorithms and Classifier Systems. In Rawlings, G. (ed.).
- Gantovnik, V. B., Anderson-Cook, C. M., Gürdal, Z., und Watson, L. T. (2003) A genetic algorithm with memory for mixed discrete-continuous design optimization. *Computers & Structures*, 81(20), 2003-2009.
- Gellens, D. und Roulin, E. (1998) Streamflow response of Belgian catchments to IPCC climate change scenarios. *Journal of Hydrology*, 210(1-4), 242-258.
- Gerdes, I., Klawonn, F., und Kruse, R. (2004) *Evolutionäre Algorithmen*, Vieweg+Teubner Verlag.
- Goldberg, D. E. und Lingle, J. R. (1985) "Alleles, Loci and the TSP" in Proceedings of the First International Conference on Genetic Algorithms and Their Applications. Hillsdale, New Jersey, In Grefenstette, J. J. (ed.), 154-159.
- Grefenstette, J., Gopal, R., Rosmaita, B., und Van Gucht, D. (1985) "Genetic Algorithms for the TSP" in Proceedings of the First International Conference on Genetic Algorithms and Their Applications. Hillsdale, New Jersey: Lawrence Erlbaum., 160-165.
- Grefenstette, J. J. (1987) Incorporating problem specific knowledge into genetic algorithms. *Genetic algorithms and simulated annealing*, 4, 42-60.
- Grosan, C. (2007) *Hybrid Evolutionary Algorithms*, Springer.
- Gwiazda, T. D. (2006) *Genetic Algorithms Reference Volume 1 Crossover for single-objective numerical optimization problems*, Tomasz Gwiazda.
- Gwiazda, T. D. (2007) *Genetic Algorithms Reference Volume 2 Mutation operator for numerical optimization problems*, Tomasz Gwiazda.
- Hättenschwiler und Gachet (2009) Einführung, Konzepte der Entscheidungsunterstützung. [online] <http://diuf.unifr.ch/ds/courses/dss2002/pdf/DSS-Einfuehrung.pdf> (Zugegriffen Mai 17, 2009).
- Holland, J. (1975) *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, Michigan, Ann Arbor The University of Michigan Press.
- Huebner, C. (2003) *Entwicklung eines optimalen Hochwassersteuerungskonzeptes für das Einzugsgebiet der Erft*, Vertiefearbeit. Technische Universität Darmstadt, Fachgebiet Ingenieurhydrologie und Wasserbewirtschaftung.
- Huebner, C. (2004) *Optimale Parameterwahl für Evolutionäre Algorithmen zur Optimierung von Mischwassernetzen*, Diplomarbeit. Technische Universität Darmstadt, Fachgebiet Ingenieurhydrologie und Wasserbewirtschaftung.
- Hughes, W. (1971) Flood Control Release Optimization using Methods from Calculus.
- Huisinga, W., Liebermeister, W., und Jahnke, T. (2009) Computational Physiology. [online] <http://compphysiol.mi.fu-berlin.de/> (Zugegriffen August 26, 2009).

-
- Intel Corporation (2009a) Intel® Visual Fortran Compiler Professional Edition 11.0 for Windows In-Depth. Intel® Compilers - Intel® Software Network. [online] <http://software.intel.com/en-us/intel-compilers/> (Zugegriffen April 23, 2009).
- Intel Corporation (2009b) Intel® VTune™ Performance Analyzer. Intel® VTune - Intel® Software Network. [online] <http://software.intel.com/en-us/intel-vtune/> (Zugegriffen April 24, 2009).
- ISFD4 (2008) Statement of the 4th International Symposium on Flood Defence.
- Jairaj, P. G. und Vedula, S. (2000) Multireservoir System Optimization using Fuzzy Mathematical Programming. *Water Resources Management*, 14(6), 457-472.
- Jüpner, R. (2005) *Das kommunale Hochwassermanagementsystem: Ein Weg zur Umsetzung der zukunftsweisenden Hochwasserschutzstrategie*, Hochwassermanagement. Magdeburger Wasserwirtschaftliche Hefte.
- Jürgen Giesecke (2001) *Strategien des Hochwasserrückhaltes*, Kassel, Herkules-Verl.
- Kerber, D. (2009) Entwicklung eines multikriteriellen, hybriden Optimierungsalgorithmus für den Einsatz in der Kanalnetzsteuerung.
- Klawitter, A. (2006) Ein Modellkonzept zur integrativen Betrachtung von Urban- und Ruralhydrologie auf Einzugsgebietsebene. [online] <http://tubiblio.ulb.tu-darmstadt.de/25874/>.
- Klein, B., Schumann, A., und Pahlow, M. (2008) Hochwasserschutzplanung in Flussgebieten unter Verwendung der multivariaten Statistik am Beispiel der Unstrut. *Wasserwirtschaft*, (2008-11).
- Kofler, M. (2000) *Excel 2000 programmieren . Anwendungen erstellen mit Visual Basic für Applikationen*, Addison-Wesley.
- Kron, W. (2003) Maßnahmen zum Hochwasserschutz und zur Hochwasservorsorge. Hinweise zur DKKV-Studie "Hochwasservorsorge in Deutschland", Schriftenreihe des DKKV 29.
- Kularathna, M. und Bogardi, J. J. (1989) "A stochastic dynamic programming based multiobjective analysis of a water resources system using a simplified configuration" in Bogardi, JJ (1989): Proc. Int. Seminar Workshop Conflict analysis in reservoir management, S., 445–454.
- Kularathna, M. und Bogardi, J. J. (1990) "Simplified system configurations for stochastic dynamic programming based optimization of multireservoir systems" in Goulter, IC [Hrsg.]: Water resource systems application. Proceedings of the International Symposium on Water Resources Systems Application, June., 12–19.
- Larranaga, P., Kuijpers, C. M. H., und Murga, R. H. (1994) Tackling The Travelling Salesman Problem With Evolutionary Algorithms: Representations And Operators. [online] <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.4969> (Zugegriffen Mai 25, 2009).
- Larranaga, P., Kuijpers, C., Murga, R., Inza, I., und Dizdarevic, S. (1999) Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators. *Artificial Intelligence Review*, 13(2), 129-170.

-
- Lidd, M. L. (1991) Traveling salesman problem domain application of a fundamentally new approach to utilizing genetic algorithms. Research sponsored in part by Air Force Office of Scientific Research and Office of Naval Research, Contract F, 4920.
- Löffelmann, K. (2006) *Microsoft Visual Basic 2005 - Das Entwicklerbuch*, Microsoft Press Deutschland.
- Lohr, H. (2001) *Simulation, Bewertung und Optimierung von Betriebsregeln für wasserwirtschaftliche Speichersysteme*, Darmstadt, Technische Universität Darmstadt, Mitteilungen des Instituts für Wasserbau und Wasserwirtschaft. Heft 118.
- Louden, K. C. (1998) *Programmiersprachen. Grundlagen, Konzepte, Entwurf*, Redline GmbH.
- Louden, K. C. (2002) *Programming Languages : Principles and Practice*, Thomson Press.
- Lüderitz, V. und Jüpner, R. (2009) "Renaturierung von Fließgewässern" in Renaturierung von Ökosystemen in Mitteleuropa., 95-124. [online] http://dx.doi.org/10.1007/978-3-8274-2161-6_4 (Zugegriffen Januar 21, 2010).
- MacDonald, M. (2006) *The Book of Visual Basic 2005: .NET Insight for Classic VB Developers*, No Starch Press.
- Mehler und Ostrowski (1998) Herleitung des hydrologischen Speicherkonzeptes für lineare und bereichsweise lineare Übertragungsfunktionen. [online] http://wiki.bluemodel.org/images/1/17/Mehler_1998_Mathematische_Herleitung.pdf.
- Mehler, R. und Leichtfuß, A. (2001) "Application of Evolutionary Algorithms to Achieve Optimal Configuration of Urban Drainage Systems" in Urban Drainage Modeling: Proceedings of the Specialty Symposium Held in Conjunction With the World Water and Environmental Resources Congress. Orlando, American Society of Civil Engineers, 847-855.
- Mehler, R. und Leichtfuß, A. (2003) Simulationsgestützte Optimierung von Stadtentwässerungsnetzen mit Hilfe von Evolutionsstrategien. *Wasser und Abfall*, (6), 35-39.
- Meyer, D. (2002) MOMBES Multiobjective Modelbased Evolution Strategy.
- Meyer-Zurwelle, J. (1975) *Optimale Abgabestrategien für Hochwasserspeichersysteme*, Karlsruhe, Inst. Wasserbau III, Univ. Karlsruhe.
- Michalewicz, Z. (1996) *Genetic algorithms+ data structures= evolution programs*, Springer.
- Ministerium fuer Umwelt und Naturschutz, Landwirtschaft und Verbraucherschutz NRW (2009) Gewässersteckbrief Erft. [online] http://www.erft.nrw.de/Wasserkoerpersteckbriefe/Gewaessersteckbrief_Erft/index.jsp (Zugegriffen November 20, 2009).
- Moscato, P. (1989) On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Caltech Concurrent Computation Program, C3P Report, 826, 1989.
- Muschalla, D. (2006) Evolutionäre multikriterielle Optimierung komplexer wasserwirtschaftlicher Systeme.

-
- Nürnberg, H. und Beyer, H. (1997) The dynamics of evolution strategies in the optimization of traveling salesman problems.
- Oliver, I. M., Smith, D. J., und Holland, J. R. C. (1987) "A Study of Permutation Crossover Operators on the TSP, Genetic Algorithms and their Applications" in Proceedings of the Second International Conference on Genetic Algorithms., 224-230.
- Olsen, J., Beling, P., und Lambert, J. (2000) Dynamic models for floodplain management. *Journal of Water Resources Planning and Management-Asce*, 126(3), 167-175.
- Opsys, Inc. (2009) Linux Trace Toolkit. Linux Trace Toolkit. [online]
<http://www.opsys.com/LTT/> (Zugegriffen April 24, 2009).
- Ormsbee, L. E., Houck, M. H., und Delleur, J. W. (1987) Design of Dual-Purpose Detention Systems using Dynamic Programming. *Journal of Water Resources Planning and Management*, 113(4), 471-484.
- Ostrowski (1977) Optimale Entnahmesteuerung von Hochwasserspeichern im Verbund.
- Ostrowski und Wolf (1984) Der Einfluß der zeitlichen Untergliederung auf die Parameterschätzung in hydrologischen Modellen. Sonderdruck aus Deutsche Gewässerkundliche Mitteilungen, 28.1984, H. 2, S. 37-40.
- Ostrowski, M. W. (1992) Ein universeller Baustein zur Simulation hydrologischer Prozesse. *Wasser & Boden*, 44(11), 755-760.
- Ostrowski, M. W., Bach, M., Heusch, S., Huebner, C., Fröhlich, F., und Reußner, F. (2009) Hydrologische Grundlagen - BlueM. [online]
chrome://iatab/content/reloaded.html?url=http://130.83.196.220/bluem/wiki/index.php/Hydrologische_Grundlagen (Zugegriffen Oktober 16, 2009).
- Ostrowski, M. W., Lohr, H., und Mehler, R. (2002) Dokumentation des Speicherbetriebsmodells TALSIM 2.0.
- Ostrowski, M. W., Mehler, R., Lohr, H., Bente, S., und Lempert, M. (1999) Ein universeller, nicht linearer Speicherbaustein zur Simulation hydrologischer Systeme.
- Otero, J. M., Labadie, J. W., Haurert, D. E., und Daron, M. S. (1995) "Optimization of managed runoff to the St. Lucie Estuary" in *Water Resources Engineering.*, 1506-1510.
- Papageorgiou, M. (1996) *Optimierung: statische, dynamische, stochastische Verfahren für die Anwendung*, München, Oldenbourg.
- Patt, H. (2001) *Hochwasser-Handbuch : Auswirkungen und Schutz*, Berlin [u.a.], Springer.
- Pfister, G. F. (1998) *In search of clusters*, Upper Saddle River, NJ, Prentice Hall.
- Pflug, W. und Schneider, T. (1998) "Erftverlegungen" in Braunkohlentagebau und Rekultivierung. *Landschaftsökologie-Folgenutzung-Naturschutz*. Springer, Berlin, 373 - 378.
- Plate, E. J. (2008) "Examples of mainstreaming flood preparedness in local communities" in 4th International Symposium on Flood Defence.

-
- Plate, E. J. und Merz, B. (2001) *Naturkatastrophen: Ursachen, Auswirkungen, Vorsorge*, Schweizerbart.
- Pohlheim, H. (1998) *Entwicklung und systemtechnische Anwendung Evolutionärer Algorithmen*, Aachen, Shaker.
- Pohlheim, H. (1999) *Evolutionäre Algorithmen*, Springer.
- Popper, K. (1974) The rationality of scientific revolutions. *Problems of scientific revolution*, 772–101.
- Pratt, T. und Zelkowitz, M. (1999) *Programmiersprachen. Design und Implementierung*, Prentice Hall.
- Rauber, T. und Rüniger, G. (2000) *Parallele und verteilte Programmierung.*, Springer, Berlin.
- Ravens, T. (2004) *Wissenschaftlich mit Excel arbeiten*, Pearson Studium.
- Rechenberg, I. (1994) *Evolutionsstrategie '94*, 1, Frommann-Holzboog.
- Rechenberg, I. (1973) *Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Stuttgart, Frommann-Holzboog.
- Reis, L., Walters, G., Savic, D., und Chaudhry, F. (2005) Multi-Reservoir Operation Planning Using Hybrid Genetic Algorithm and Linear Programming (GA-LP): An Alternative Stochastic Approach. *Water Resources Management*, 19(6), 831-848.
- Richter, B., Baumgartner, J., Powell, J., und Braun, D. (1996) A Method for Assessing Hydrologic Alteration within Ecosystems. *Conservation Biology*, 10(4), 1163-1174.
- Schielen, R. und Barneveld, H. (2006) *Die Maas im 21. Jahrhundert—Integrale Lösungsansätze zur Hochwasserproblematik*, Ministerium für Verkehr und Wasserwirtschaft, Maastricht, Niederlande.
- Schuetze, M., Butler, D., und Beck, M. (2002) *Modelling, Simulation and Control of Urban Wastewater Systems*, London, Springer.
- Schütz, K. (2007) *Optimierung der Retentionspotenziale im Einzugsgebiet der Erft*, Diplomarbeit. Unveröffentlicht.
- Schwefel, H. P. (1987) *Collective phenomena in evolutionary systems*, Dekanat Informatik, Univ.
- Schwefel, H. P. (1994) On the evolution of evolutionary computation. *Computational Intelligence: Imitating Life*, 116–124.
- Schwefel, H. (1995) *Evolution und Optimum Seeking*, Chichester, John Wiley & Sons Inc.
- Schwefel, H. (1981) *Numerical optimization of computer models (Translation of 1977 Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, Chichester ; New York, Wiley.
- Schwefel, H. (1974) *Numerische Optimierung von Computer-Modellen (PhD thesis)*,

-
- Schwefel, H. (1977) *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie : mit einer vergleichenden Einführung in die Hill-Climbing- und Zufallsstrategie*, Basel; Stuttgart, Birkhäuser.
- Sharif, M. und Wardlaw, R. (2000) Multireservoir Systems Optimization Using Genetic Algorithms: Case Study. *Journal of Computing in Civil Engineering*, 14(4), 255-263.
- Sharma (2008) Inflow prediction and optimal operation of reservoir system during flood by the combined application of ANN and different Optimization techniques.
- Simonovic, S. (1992) Reservoir Systems-Analysis - Closing Gap between Theory and Practice. *Journal of Water Resources Planning and Management-Asce*, 118(3), 262-280.
- Sören Blom (2003) Memetische Algorithmen - Ein überblick.
- Srinivas, N. und Deb, K. (1995) Multiobjective Function Optimization using Nondominated Sorting Genetic Algorithms. *Evolutionary Computation Journal*, 2(3), 221-248.
- Staatliches Umweltamt Köln (2003) Die Europäische Wasserrahmenrichtlinie - Umsetzung im Einzugsgebiet der Erft. [online] <http://www.erft.nrw.de/hyco/e/erft/main/tr/Frame.html> (Zugegriffen Dezember 14, 2003).
- Syrjakow, M. (2005) Simulationstechnik. [online] http://ces.univ-karlsruhe.de/goethe/syrjakow/simtech_vorlesung/vl_material/modelloptimierung.pdf (Zugegriffen Oktober 1, 2009).
- Turgeon, A. und Charbonneau, R. (1998) An Aggregation-Disaggregation Approach to Long-Term Reservoir Management. [online] <http://www.agu.org/pubs/crossref/1998/98WR02608.shtml> (Zugegriffen Oktober 11, 2009).
- Van Veldhuizen, D. (1999) Multiobjective Evolutionary Algorithms: Classifications Analyses and New Innovations.
- Veldhuizen, D. und Lamont, G. (2000) Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation*, 8(2), 125-147.
- Vriend, H. J. und Dijkman, J. P. M. (2003) "A new method of decision support to river flood management" in *Proceedings of the 1st International Yellow River Forum.*, 21–24.
- Winkler, N. S. (2006) Optimierung der Steuerung von Hochwasserrückhaltebeckensystemen.
- Wolbring, F. (1997) *Wissensbasierte Methoden für den Betrieb von Talsperren*, Bochum, Univ.
- Wong, M. L. (2009) "Parallel multi-objective evolutionary algorithms on graphics processing units" in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*. Montreal, Québec, Canada, ACM, 2515-2522. [online] <http://portal.acm.org/citation.cfm?id=1570354> (Zugegriffen Oktober 2, 2009).
- Young, M. D. (1992) Sustainable investment and resource use: Equity, environmental integrity and economic efficiency. *Man and the Biosphere Series*, 9, 171.

Zitzler, E., Laumanns, M., und Bleuler, S. (2002) “Tutorial on Evolutionary Multiobjective Optimization” in Workshop on Multiple Objective Metaheuristics. Paris, Springer-Verlag.

Anlagen

Anlage A	Repräsentanzen, Operatoren und Verlauf eines Rundreiseproblems	168
Anlage A - 1	Repräsentanzen des Rundreiseproblems	168
Anlage A - 2	Reproduktionsoperatoren	170
Anlage A - 3	Mutationsoperatoren	172
Anlage A - 4	Verlauf einer Optimierung des Rundreiseproblems	175
Anlage B	Reproduktions- und Mutationsoperatoren zur Lösung des Hochwasserspeicheroptimierungsproblems	176
Anlage B - 1	Reproduktionsoperatoren	176
Anlage B - 2	Mutationsoperatoren	177
Anlage C	Klassifizierung des memetischen Algorithmus	179
Anlage D	Erftverlegungen	182
Anlage E	Hochwasserschutzmaßnahmen	184
Anlage E - 1	Wasserrückhalt und Abflussleistung	184
Anlage E - 2	Hochwasserschutz in der Bauleitplanung	184
Anlage E - 3	Gesetzliche Verankerung des HW Schutz in der Bauleitplanung (Patt und Bechteler, 2001)	185
Anlage F	Systemelemente des Erft-Optimierungssystems	186
Anlage G	Steuerungsdateien des BlueM.Opt Optimierungswerkzeuges	189
Anlage G - 1	Steuerungsdatei zur kombinatorischen Optimierung	189
Anlage G - 2	Steuerungsdatei mit den Optimierungszielen	190
Anlage G - 3	Steuerungsdatei mit Optimierungsparametern	191
Anlage G - 4	Steuerungsdatei mit Modellparametern	192
Anlage H	Einstellungen zur Kontrolle der Algorithmen	193
Anlage H - 1	Steuerparameter des kombinatorischen Algorithmus	194
Anlage H - 2	Steuerparameter des Memetischen Evolutionären Algorithmus	194
Anlage H - 3	Steuerparameter des reellwertigen Algorithmus	194

Anlage A Repräsentanzen, Operatoren und Verlauf eines Rundreiseproblems

Anlage A - 1 Repräsentanzen des Rundreiseproblems

Die Unterschiede der Repräsentanzen für das Rundreiseproblem (Abbildungsvorschriften) seien hier anhand der der Tour:

$$3 - 2 - 4 - 1 - 7 - 5 - 8 - 6$$

kurz erläutert. Diese wird so interpretiert, dass die Reise von der Stadt Nummer 3 nach Stadt Nummer 2 usw. bis zum Ausgangspunkt 3 führt.

Binary Representation

Bei der Binary Representation schlägt Lidd (1991) vor, die Tour durch Binärzahlen auszudrücken:

$$T_B = (010\ 001\ 011\ 000\ 110\ 100\ 111\ 101)$$

Diese Darstellung eignet sich insbesondere für die Optimierung mittels genetischer Algorithmen, da diese ohnehin Binärzahlen für die Genotypen verwenden.

Path Representation

Die Path Representation kann am besten nachvollzogen werden. Die Tour wird in der bereits bekannten Form notiert:

$$T_P = (3\ 2\ 4\ 1\ 7\ 5\ 8\ 6)$$

Ordinal Representation

Bei der Ordinal Representation (Grefenstette, J. et al., 1985) wird die Tour durch eine Liste von n Städten repräsentiert. Hier ist das i -te Element der Liste eine Zahl aus dem Bereich zwischen 1 und $n - i + 1$. Die Tour wird mit einer Liste hinterlegt:

$$L = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$$

Als Erstes wird die Stadt mit der Nummer 3 angefahren, diese steht an **dritter** Stelle. Das ergibt die erste Ziffer der ordinalen Tour T_O . Die Stadt mit der Nummer 3 wird dann aus der Liste L entfernt. Danach wird die Stadt 2 angefahren, diese steht an Stelle **2** und wird aus der Liste genommen. Dementsprechend wird die nächste Stadt mit der Nr. 4 gleichfalls durch eine **2** repräsentiert:

$$T_O = (3\ 2\ 2\ 1\ 3\ 1\ 2\ 1)$$

Adjacency Representation

Ebenfalls in der Arbeit von Grefenstette, J. et al. (1985) wird Adjacency Representation vorgeschlagen. Auch hier wird die Tour durch eine Liste von n Städten repräsentiert. Die Stadt j steht an Position i dann, und nur dann, wenn die Tour von der Stadt i zur Stadt j führt. Die Tour wird in folgender Form notiert:

$$T_A = (7\ 4\ 2\ 1\ 8\ 3\ 5\ 6)$$

Matrix Representation

Der Vollständigkeit halber sei an dieser Stelle auch die Matrix Representation von Fox und McMahon (1987) erläutert. Die Tour wird als Matrix dargestellt, in der das Element in der Reihe i und Spalte j eine 1 ist, falls und nur falls, in der Tour die Stadt i vor der Stadt j besucht wurde. Die bereits bekannte Tour:

$$3 - 2 - 4 - 1 - 7 - 5 - 8 - 6$$

Folglich hat die Matrix Representation diese Form:

$$T_M = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Diskussion

Die Binary und die Matrix Repräsentationen eignen sich nicht für die Abbildung des Pfades und die anschließende Optimierung mit einer Evolutionsstrategie, da der Pfad binär dargestellt wird. Sie sind allerdings prädestiniert für die genetische Optimierung, da ihre Mutations- und Reproduktionsalgorithmen auf binär codierte Probleme spezialisiert sind.

Bezüglich der verbliebenen drei Repräsentanzen spricht für die Path Representation die Transparenz der Notation. Die Adjacency Representation und die Ordinal Representation machen auf den ersten Blick einen etwas umständlichen Eindruck. Aber auch diese Abbildungsvorschriften haben ihre Vorteile. Beispielsweise erlaubt die Adjacency Representation eine Analyse der Hyperebenen des n -dimensionalen Lösungsraums. Hinsichtlich der Effizienz zugehöriger Reproduktionsoperatoren unterliegen sie aber der

Path Representation. Das liegt vorwiegend daran, dass gute Subtouren relativ häufig wieder verworfen werden (Oliver et al., 1987; Grefenstette, J. et al., 1985; Michalewicz, 1996, zitiert nach: Larranaga et al., 1994).

Nicht nur, weil die Notation der Path Representation schlicht einleuchtet, sondern auch weil sie sich als effizient erwiesen hat, wurden für diese von vielen Autoren weitere Reproduktions- und Mutationsoperatoren entwickelt (Larranaga u. a., 1999).

Anlage A - 2 Reproduktionsoperatoren

Im Folgenden werden die Reproduktionsoperatoren "Partially Mapped Crossover" und "Order Crossover" beschrieben. Die Operatoren unterscheiden sich hinsichtlich des Erhalts der Position und des Erhalts der Reihenfolge der Städte. Der Order Crossover Operator berücksichtigt vorwiegend die Reihenfolge, während der Partially Mapped Crossover auch die Position der Städte zu erhalten versucht.

Reproduktionsoperator "Partially Mapped Crossover" (PMX)

Der Reproduktionsoperator *Partially Mapped Crossover* wurde von Goldberg und Lingle (1985) vorgeschlagen. Hierbei wurden Informationen bezüglich der Reihenfolge und bezüglich der Werte an die Nachkommen übertragen. Ein Teil der Tour des Elter wird auf einen Teil des anderen Elter übertragen. Die übrigen Teile des Strings werden ausgetauscht. Wie nehmen an, es sind die folgenden beiden Eltern vorhanden:

(1 2 3 4 5 6 7 8) und

(3 7 5 1 6 8 2 4).

Der PMX-Operator generiert die Nachfahren folgendermaßen. Zunächst werden zufällig zwei Schnittpunkte (Cut Points) bei beiden Eltern an der gleichen Position definiert. Angenommen die Schnittpunkte liegen zwischen dem dritten und dem vierten Element und zwischen dem Sechsten und Siebten. Beispielsweise so:

(1 2 3 | 4 5 6 | 7 8) und

(3 7 5 | 1 6 8 | 2 4).

Der Substring zwischen den beiden Schnittpunkten wird "Mapping Section" genannt. In diesem Beispiel wird die $4 \leftrightarrow 1$, $5 \leftrightarrow 6$, $6 \leftrightarrow 8$ abgebildet. Nun wird die Mapping Section

des ersten Elter auf den zweiten Nachfahren übertragen und die des zweiten Elter auf den ersten Nachfahren:

Nachfahre 1: $(x\ x\ x\ | 1\ 6\ 8\ | x\ x)$ und

Nachfahre 2: $(x\ x\ x\ | 4\ 5\ 6\ | x\ x)$.

Danach wird der Nachfahre i ($i = 1,2$) mit den Elementen des i ten Elter aufgefüllt. Falls im String des Nachfahren eine Stadt bereits vorhanden ist, wird diese entsprechend des Mappings ersetzt. Beispielsweise wäre das erste Element des ersten Nachfahren eine 1, wie im ersten Element des ersten Elter. Die 1 ist allerdings bereits im String vorhanden. Daher wird beim Mapping $4 \leftrightarrow 1$ entsprechend die 1 durch die 4 ersetzt. Das zweite, dritte und siebte Element des Nachfahren 1 kann vom Elter 1 übernommen werden. Das letzte Element des ersten Nachfahren wäre eine 8, die allerdings bereits im String vorhanden ist. Daher wird wieder den Mappings folgend $8 \leftrightarrow 6$ und $6 \leftrightarrow 5$ abgebildet und an die letzte Stelle eine 5 geschrieben:

Nachfahre 1: $(4\ 2\ 3\ | 1\ 6\ 8\ | 7\ 5)$ und

Nachfahre 2: $(3\ 7\ 8\ | 4\ 5\ 6\ | 2\ 1)$.

Ein Vorteil dieses Verfahrens ist, dass einige Elemente und ggf. auch deren Reihenfolge der Eltern erhalten bleiben.

Reproduktionsoperator "Order Crossover" (OX)

Der Reproduktionsoperator *Order Crossover* wurde von Davis (1985) entwickelt. Dieser Operator macht sich die Eigenschaft der Path Representation zunutze, dass lediglich die Reihenfolge der Städte und nicht ihre Position relevant ist. Ein Nachfahre wird erstellt, indem eine Subtour eines Elter selektiert wird und die relative Reihenfolge des anderen Elter erhalten wird. Angenommen die folgenden zwei Eltern liegen vor:

$(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$ und

$(3\ 7\ 5\ 1\ 6\ 8\ 2\ 4)$;

Des Weiteren wird angenommen, dass der erste Schnittpunkt zwischen dem zweiten und dritten Element und der zweite Schnittpunkt zwischen dem fünften und sechsten Element liegt:

$(1\ 2\ | 3\ 4\ 5\ | 6\ 7\ 8)$ und

(3 7 | 5 1 6 | 8 2 4);

Die Nachfahren werden folgendermaßen generiert. Zunächst wird die Subtour zwischen den Schnittpunkten in die Nachfahren kopiert:

Nachfahre 1: (x x | 3 4 5 | x x x) und

Nachfahre 2: (x x | 5 1 6 | x x x).

Beginnend nach dem zweiten Schnittpunkt eines Elter werden die übrigen Elemente der dort vorliegenden Reihenfolge in den anderen Nachfahren kopiert. Bereits vorhandene Städte werden übersprungen. Wenn das Ende des Strings erreicht ist, wird mit dem ersten Element fortgefahren. In diesem Beispiel sieht das folgendermaßen aus:

Nachfahre 1: (1 6 | 3 4 5 | 8 2 7) und

Nachfahre 2: (3 4 | 5 1 6 | 7 8 2).

Beide Operatoren sind in der Lage gültige Permutationen des Rundreiseproblems zu reproduzieren. Nachträgliche Validitätsprüfungen und Korrekturen sind nicht erforderlich.

Anlage A - 3 Mutationsoperatoren

Zur Mutation des Genotyps wurden die vier Operatoren "Simple Inversion Mutation", "Translocation", "Translocation" und "Exchange Mutation Operator" implementiert. Sie unterscheiden sich hinsichtlich der Mutationsstärke und der Methode, wie die Reihenfolge erhalten bzw. verworfen wird.

Mutationsoperator "Simple Inversion Mutation" (SIM)

Der *simple inversion mutation operator* von J.H. Holland (1975) und Grefenstette (1987) setzt zufällig zwei Schnittpunkte in die Strings und der mittlere Substring wird invertiert. Beispielsweise mutiert die folgende Tour mit den Schnittpunkten zwischen der zweiten und dritten Stadt und der fünften und sechsten Stadt

(1 2 | 3 4 5 | 6 7 8)

zu

(1 2 | 5 4 3 | 6 7 8).

Mutationsoperator "Translocation" (3Opt)

Beim *translocation operator* von Nürnberg und Beyer (1997) wird der String mit zwei zufällig gesetzten Schnittpunkten in drei Subtouren zerlegt. Um den Nachfahren zu mutieren, werden die drei Teile in zufälliger Reihenfolge wieder zusammengesetzt. Für die Subtouren wird Bernoulli verteilt (Null-Eins-Verteilung) und die Inverse verwendet. Angenommen es liegt folgende Tour und ihre Schnittpunkte vor:

$$(1\ 2\ | 3\ 4\ 5\ | 6\ 7\ 8)$$

Zunächst werden die Subtouren in beliebiger Reihenfolge neu arrangiert:

$$(6\ 7\ 8\ | 1\ 2\ | 3\ 4\ 5)$$

Im nächsten Schritt werden die beiden äußeren Touren invertiert:

$$(8\ 7\ 6\ | 1\ 2\ | 5\ 4\ 3).$$

Mutationsoperator "Translocation" (nOpt)

In dieser Arbeit wurde obiger Translocation Operator *3-Opt* erweitert. Der Operator wurde flexibler formuliert, sodass der String nicht nur in drei Teile zerlegt werden konnte, sondern in n Teile. Dementsprechend die Namensgebung *n-Opt*. Angenommen es liegt folgende Tour und ihre Schnittpunkte vor:

$$(1\ 2\ | 3\ 4\ | 5\ 6\ | 7\ 8)$$

Wie beim Operator *3-Opt* werden die Subtouren in beliebiger Reihenfolge neu arrangiert:

$$(3\ 4\ | 7\ 8\ | 1\ 2\ | 5\ 6)$$

Im nächsten Schritt werden die Subtouren Bernoulli verteilt invertiert:

$$(3\ 4\ | 8\ 7\ | 2\ 1\ | 5\ 6)$$

Mutationsoperator "Exchange Mutation Operator" (EM)

Der *exchange mutation operator* (Banzhaf, 1990) wählt zufällig zwei Städte der Tour und vertauscht sie. Angenommen es liegt folgende Tour vor:

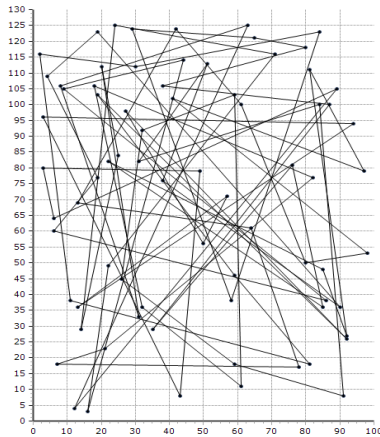
$$(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$$

Die dritte und fünfte Stadt werden ausgewählt. Dann sieht die mutierte Tour wie folgt aus:

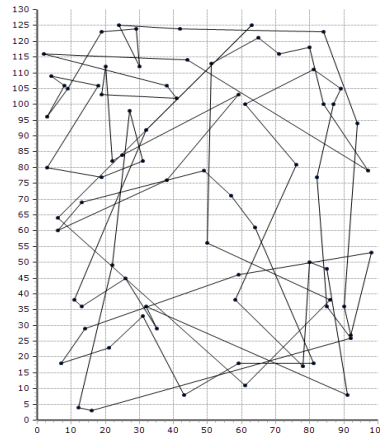
$$(1\ 2\ 5\ 4\ 3\ 6\ 7\ 8)$$

Beyer (1992) wiederholt die exchange mutation mehrfach. Er führt einen Kontrollparameter s ein, der die Anzahl der Wiederholungen angibt. Hierbei kann jedes Individuum seinen eigenen Wert für s besitzen. Er wird aus den Werten für s der Eltern ermittelt. Zu Beginn sind die Werte für s relativ hoch. Während der Optimierung geht der Wert s gegen 1. Diese Methode wurde vom *Dn – Vektor* von Hans-Paul Schwefel (1974) abgeleitet.

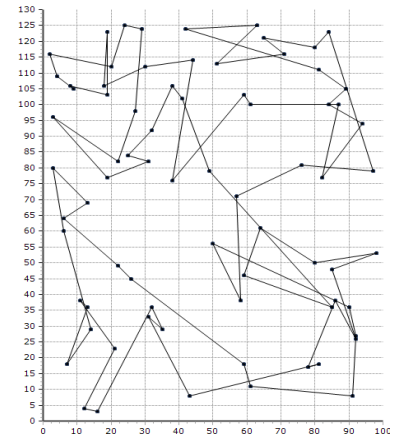
Anlage A - 4 Verlauf einer Optimierung des Rundreiseproblems



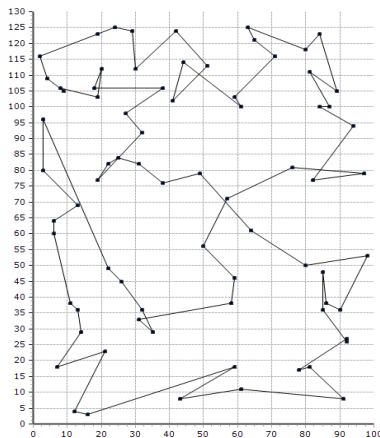
Generation: 1, Qualität: 4191



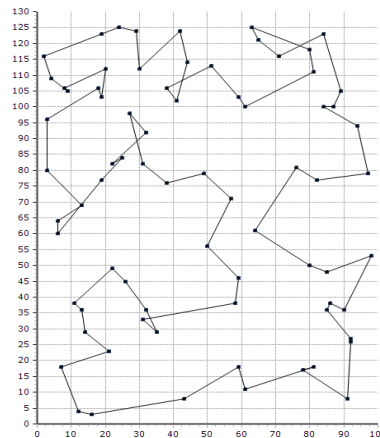
Generation 100, Qualität: 2053



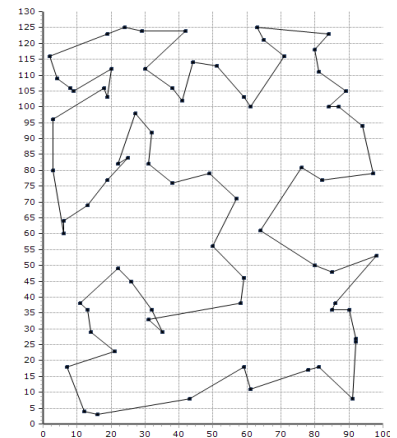
Generation 1.000, Qualität: 1404



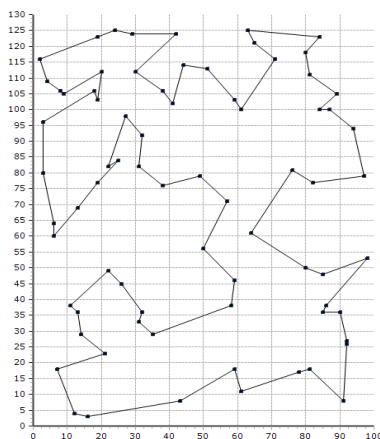
Generation 3.000, Qualität: 995



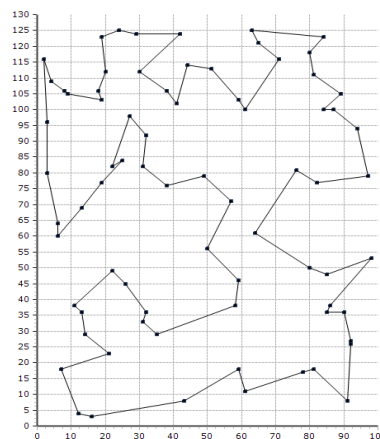
Generation 5.000, Qualität: 850



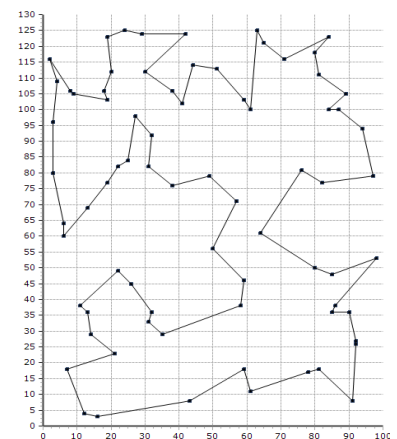
Generation 10.000, Qualität: 807



Generation 13.000, Qualität: 798



Generation 16.000, Qualität: 787



Generation 19.000, Qualität: 781

Städte: 75, Kombinationen: $1,24 \cdot 10^{109}$, Selektionsdruck: $1/4$, Evaluierungen: 400.000

Anlage B Reproduktions- und Mutationsoperatoren zur Lösung des Hochwasserspeicheroptimierungsproblems

Anlage B - 1 Reproduktionsoperatoren

Reproduktionsoperator: „One Point Crossover“ (OPC) und „Two Point Crossover“ (TPC)

Der One Point Crossover Operator und der folgende Two Point Crossover Operator wurden von Holland (1975) und De Jong (1975) entwickelt und erprobt. Die Operatoren wurden damals von Holland für die genetischen Algorithmen entwickelt, welche auf binären Strings basieren. Im Bereich der genetischen Algorithmen wird häufig der aus der Genetik stammende Begriff „Allel“ verwendet. Dieser bezeichnet eine mögliche Ausprägung eines Gens, das sich an einem bestimmten Ort (Lokalität) auf einem Chromosom befindet. Bei diesem One Point Crossover Operator wird zufällig ein Schnittpunkt gesetzt und die Allele der Nachfahren bis zu diesem Schnittpunkt mit den Genen des einen Vorfahren und nach dem Schnittpunkt mit den Genen des anderen Vorfahren bestückt. Angenommen es liegen folgende Eltern vor:

$$(1\ 3\ 0\ | \ 2\ 1\ 2\ 1\ 3),$$

$$(3\ 0\ 1\ | \ 0\ 1\ 0\ 3\ 1).$$

Der Schnittpunkt wird nach dem dritten Allel gesetzt. Die Nachfahren sind entsprechend:

$$(1\ 3\ 0\ | \ 0\ 1\ 0\ 3\ 1),$$

$$(3\ 0\ 1\ | \ 2\ 1\ 2\ 1\ 3).$$

Verallgemeinert wird folgende Form verwendet:

$$\begin{array}{l} a_1 \dots a_{i-1} \ | \ a_i \dots a_j \\ b_1 \dots b_{i-1} \ | \ b_i \dots b_j \end{array} \rightarrow \begin{array}{l} a_1 \dots a_{i-1} \ | \ b_i \dots b_j \\ b_1 \dots b_{i-1} \ | \ a_i \dots a_j \end{array}$$

Für den Reproduktionsoperator „Two Point Crossover“ wird die Operation entsprechend erweitert:

$$\begin{array}{l} a_1 \dots a_{i-1} \ | \ a_i \dots a_{j-1} \ | \ a_j \dots a_l \\ b_1 \dots b_{i-1} \ | \ b_i \dots b_{j-1} \ | \ b_j \dots b_l \end{array} \rightarrow \begin{array}{l} a_1 \dots a_{i-1} \ | \ b_i \dots b_{j-1} \ | \ a_j \dots a_l \\ b_1 \dots b_{i-1} \ | \ a_i \dots a_{j-1} \ | \ b_j \dots b_l \end{array}$$

Reproduktionsoperator: „k-Point Crossover“ (kPC)

Die „One Point Crossover“ und „Two-Point-Crossover“ Operatoren sind letztlich Spezialfälle des „k-Point-Crossover“ oder „Multi-Point-Crossover“. Zunächst wird beim Multi-

Point-Crossover eine Anzahl von Schnittpunkten gleich verteilt zufällig gesetzt. Die Schnittpunkte werden ihrer Position im Path nach sortiert. Danach werden die Allele innerhalb aufeinanderfolgender Schnittpunkte zwischen den Elternindividuen zur Bildung zweier Nachkommen ausgetauscht (Pohlheim, 1998 und 1999).

Reproduktionsoperator: "Uniform Crossover" UX

Ein Nachkomme wird gebildet, indem an jedem Standort bernulliverteilt die Maßnahme des ersten oder zweiten Vorfahren gewählt wird. Angenommen, es liegen folgende beiden Eltern vor:

(3 | 1 | 3 | 4 | 2 | 4 | 1 | 1) und

(1 | 4 | 2 | 1 | 2 | 3 | 2 | 4).

Dann würde beispielsweise für einen Nachfahren als Erstes die Maßnahme 3 aus dem ersten Elter, als Zweites die Maßnahme 4 aus dem Zweiten, als Drittes die Maßnahme 2 erneut aus dem zweiten Elter usw.

(3 | 4 | 2 | 4 | 2 | 3 | 1 | 1).

Im Gegensatz zu den Crossover Operatoren werden hier erfolgreiche Genabschnitte tendenziell öfter getrennt. Die Crossover Operatoren bewahren Genabschnitte und setzen diese neu zusammen.

Anlage B - 2 Mutationsoperatoren

Mutationsoperator "Random Bit Value" (RBV)

Der Mutationsoperator *Random Bit Value* (Gwiazda, 2007) auch *Random Switch* genannt, verändert jedes Gene mit einer vorab festgelegten Wahrscheinlichkeit. Der Operator durchläuft den Genotyp sequenzielle und entscheidet mit der Wahrscheinlichkeit der Mutationsrate, ob dieses Allel verändert wird. Beispielsweise wurden hier

(1 3 0 2 1 2 1 1)

die Allele Nr. 1, 4 und 6 neu generiert zu dem Nachfahren:

(3 3 0 0 1 0 1 1).

Mutationsoperator „Gene Insertion“ (GI)

Bei der *Gene Insertion* (Gwiazda, 2007) werden nicht einzelne Allele betrachtet, sondern Unterabschnitte des Genotyps. Es wird zunächst ein Bereich festgelegt der mutiert werden soll:

$$(1\ 3\ 0\ 2\ | \ 1\ 2\ 1\ | \ 1)$$

Dieser Teil des Genoms wird neu erstellt zu dem Nachfahren,

$$(1\ 3\ 0\ 2\ | \ 0\ 3\ 2\ | \ 1),$$

wobei die übrigen Bereiche unverändert bleiben.

Auch bei diesem Mutationsoperator wird vorab eine Mutationswahrscheinlichkeit festgelegt. Allerdings definiert diese nicht die Wahrscheinlichkeit mit der ein Allel oder ein Bereich mutiert wird, sondern die Größe des Unterabschnittes, der mutiert werden soll.

Anlage C Klassifizierung des memetischen Algorithmus

Eine erste Klassifizierung wurde in Kapitel 2.4.1 „Klassifizierung der Optimierungsalgorithmen“ vorgenommen. In diesem Abschnitt werden die Algorithmen klassifiziert die zur der Familie der evolutionären Algorithmen gehören. Wie bereits erwähnt, unterscheidet man bei den evolutionären Algorithmen grundsätzlich in genetische und evolutionsstrategische Verfahren. Innerhalb dieser beiden Klassen kann differenziert werden zwischen:

- Hybrid Genetischen Algorithmen (HGA),
- Memetischen Algorithmen (MA),
- Parallelen Algorithmen (PGA),
- Dynamisch Parametrischen Algorithmen (DPGA),
- Mikro Genetischen Algorithmen (μ GA) und
- Parallel Memetischen Algorithmen (PMA).

Allerdings ist die eindeutige Klassifizierung des anvisierten Algorithmus nur schwer möglich. Aufgrund der ständigen Weiterentwicklung und der Unzahl an Algorithmen ist eine Klassifizierung kaum zu leisten ist (Moscato, 1989). Dennoch sollen hier einige Typen von Algorithmen und deren Eigenschaften kurz beleuchtet werden, um an diesem Beispiel später den entwickelten Algorithmus zu diskutieren.

Hybride Algorithmen nutzen Synergieeffekte durch die Nutzung zweier oder mehr Algorithmen. In den letzten Jahren wurden viele solcher Algorithmen entwickelt. Die meisten, so Grosan (2007), wurden ad hoc zur Lösung bestimmter Problemstellungen entwickelt. Grosan (2007) untergliedert wie folgt:

- Evolutionary Algorithms Assisted by Evolutionary Algorithms
- Evolutionary Algorithms Assisted by Neural Networks
- Fuzzy Logic Assisted Evolutionary Algorithms
- Evolutionary Algorithms Assisted by Particle Swarm Optimization
- Evolutionary Algorithms Assisted by Ant Colony Optimization
- Evolutionary Algorithms Assisted by Bacterial Foraging
- Evolutionary Algorithms Incorporating Prior Knowledge
- Hybrid Approaches Incorporating Local Search and Others

Eine besondere Klasse entwickelte sich bei der Kombination evolutionärer Algorithmen und lokaler Suchverfahren. Diese werden auch als Memetische Algorithmen bezeichnet und berücksichtigen die Fähigkeiten und Eigenschaften, welche Individuen durch die kulturelle Evolution erhalten.

Memetische Algorithmen

Ausgehend von den Arbeiten von Popper (1974) und Cavalli-Sforza (1971) führt der Biologe Richard Dawkins in seiner Arbeit „The Selfish Gene“ (Dawkins, 1976) den Begriff des »Meme«, angelehnt an den griechischen Begriff „mimeme“ – Imitation, ein. Er bezeichnet damit Verhaltenselemente, welche sich im Gegensatz zu Genen individuell ändern können, indem sie beispielsweise durch Nachahmung erworben werden. Dabei sei das Meme als eine Art Gedankenbaustein zu verstehen, der weitestgehend unverändert weitergegeben werden kann, aber auch ähnlich wie Gene mutieren kann und durch die "Eingängigkeit", seine Speicherfähigkeit im Gehirn, der Selektion unterliegt. Dawkins nennt Beispiele für Mems wie Melodien, Gedanken, Schlagworte, Kleidermoden, die Art, Töpfe zu machen oder Bögen zu bauen und entwickelt eine Analogie zwischen genetischer und kultureller Evolution. Später stellt er selbst die Frage: „Eine Melodie ist zwar ein Mem, aber aus wie vielen Melodien besteht eine Sinfonie?“. Moscato (1989) diskutiert die Prozesse für eine kulturelle Evolution an der Entwicklung der Kampfkunst. Im Speziellen bezieht er sich auf das chinesische Kung-Fu, welches über 4.000 Jahre hinweg stetig weiterentwickelt wurde. Er führt an, dass es nicht allen Individuen, sondern lediglich den Meistern, welche im Besitz eines schwarzen Gürtels seien erlaubt sei zu lehren. Dies sei vergleichbar mit dem Selektionsprozess der evolutionären Algorithmen.

Diskussion

Eine Möglichkeit zur Klassifizierung besteht darin sich am Typ der zu optimierenden Variablen zu orientieren. Beispiele sind hier Real-Valued Genetic Algorithms oder Mixed-Integer Evolution Strategies. Der hier entwickelte Algorithmus könnte also als Mixed-Integer Evolution Strategy (MI-ES) bezeichnet werden. Dies würde darüber informieren, welche Variablentypen (Mixed-Integer für gemischt-ganzzahlig) innerhalb der Representation (Codierung) und welche Operatoren verwendet werden. Allerdings werden nicht nur die klassischen Evolutionsstrategischen Operatoren verwendet sondern auch genetische daher könnte man ihn auch als Mixed-Integer Genetic Algorithm bezeichnen.

Außerdem informiert MI-ES oder MI-GA nicht über die Funktionsweise (Prozedur) des Algorithmus.

Wie in der Einführung dieses Kapitels bereits erwähnt kann die nominell diskrete Optimierung als übergeordnete globale Suche betrachtet werden und die Optimierung der reellen Parameter als lokale Suche in einem Sublösungsraum. Dies ist vergleichbar mit einer Kombination aus biologischer Evolution und anschließender kultureller Evolution. Die Analogie kann dadurch fortgeführt werden, dass bei dem hier entwickelten Algorithmus die nominell diskrete Reproduktion und Mutation immer auf Basis der Eltern der aktuellen Generation erfolgt. Die Zeitspanne beträgt also eine Generation während bei der Optimierung der reellen Parameter, wie später deutlich wird, auf Informationen aus mehreren Generationen von Individuen zurückgegriffen wird. Daher entspricht die nominell diskrete Optimierung der generationsbasierten biologischen Evolution und die reeller Optimierung der kulturellen Evolution, welche auf Informationen der sich über viele Generationen hinweg entwickelten Kultur einer Gesellschaft zurückgreift.

Auf Grund der stimmigen Analogie wird der in Kapitel 3.2 “Methodik der Memetischen Hochwasserspeicheroptimierung“ vorgestellte und in dieser Arbeit entwickelte Algorithmus als **Memetic Evolutionary Algorithm** kurz **MEA** bezeichnet.

Anlage D Erftverlegungen

Die Erft ist im Unterlauf durch den Braunkohletagebau in der Niederrheinischen Bucht stark beeinflusst. Aufgrund einer Grundwasserabsenkung im Bereich der Erftscholle und deren Randgebiete wurde den Aquiferen, sowie den Sanden im Bereich des Hauptflözes fast vollständig das Wasser entzogen. Dies hat zur Konsequenz, dass im Einflussbereich der Sumpfungsmaßnahmen ein Grundwasseranschluss der Gewässer oder der Auen nicht mehr existiert (Staatliches Umweltamt Köln, 2003).

Die Erft musste mit Beginn des Braunkohlentieftagebaues mehrfach dem Bergbau weichen. Der Hochwasserschutz wurde nicht nur wegen der Anlieger erhöht. Sie wurde im Mittel- und Unterlauf hochwasserfrei ausgebaut, um zu gewährleisten, dass die Braunkohlentagebaue vor Hochwasser sicher waren. Zusätzlich musste sie bis zu knapp 30 m³/s Grundwasser aufnehmen, um die Tieftagebaue trocken zu halten. Daher wurde das Mittelwasserbett bis zur Mündung wiederholt umgebaut. Einerseits erfüllte die Erft für den Braunkohlenbergbau eine wichtige Funktion, nämlich die der Aufnahme der vom Bergbau geforderten Sümpfwasser. Andererseits stand sie dem Braunkohlentagebau an verschiedenen Stellen im Wege (Pflug und Schneider, 1998).

Fünf Verlegungen waren notwendig (Abbildung 8-1), um die Braunkohlentieftagebaue vor Hochwasser mit hoher Sicherheit zu schützen:

- Erste Erftverlegung bei Epprath (1941/1942)
- Zweite Erftverlegung Morken/Harff (1958)
- Dritte Erftverlegung beim RWE-Kraftwerk Frimmersdorf (1962)
- Vierte Erftverlegung Bedburg/Kaster/Harff (1971)
- Fünfte Erftverlegung Bedburg-Broich/Frimmersdorf (1973/1974)

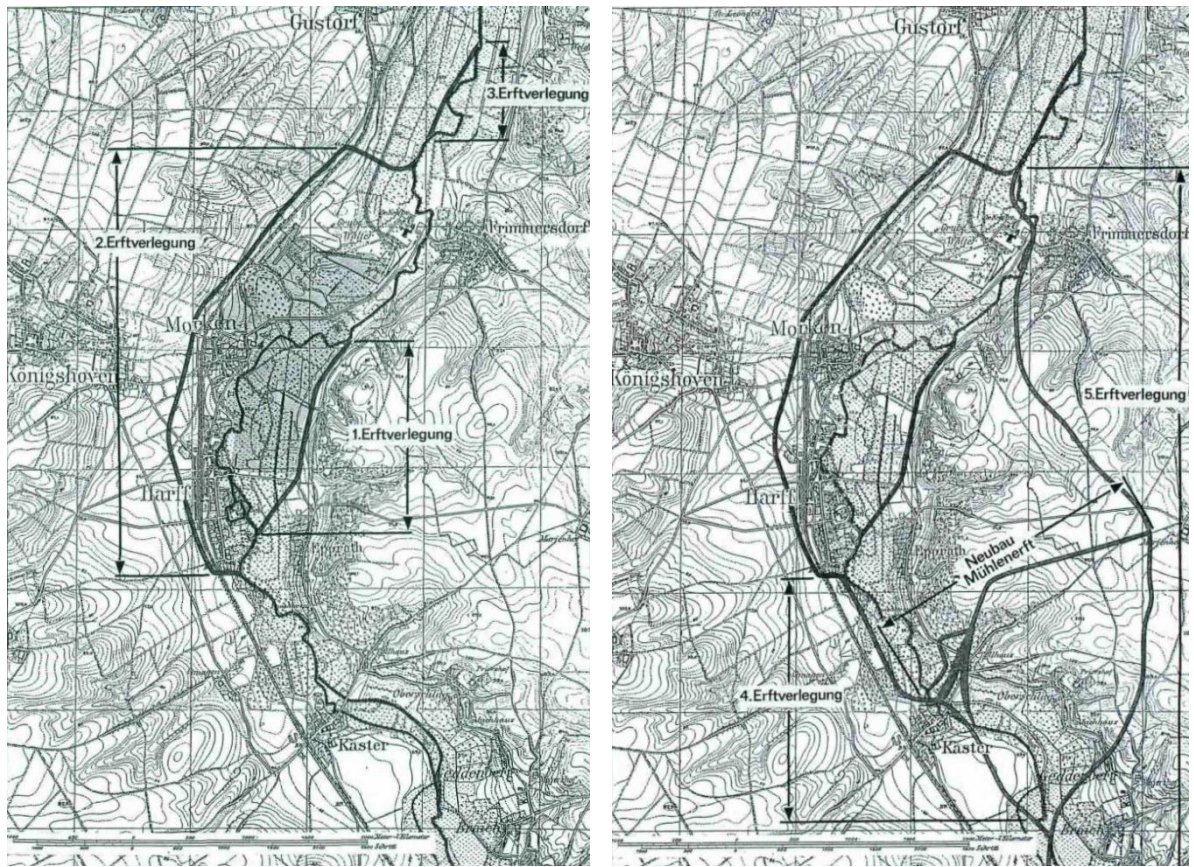


Abbildung 8-1: Ertfverlegungen (Pflug und Schneider, 1998)

Anlage E Hochwasserschutzmaßnahmen

Anlage E - 1 Wasserrückhalt und Abflussleistung

Hochwasserschutzmaßnahme	Beschreibung
Rückhalt im Abflussquerschnitt	Vergrößerung (Aufweitung, Verbreiterung) des Fließquerschnitts Reduzierung des Sohlengefälles Erhöhung der Fließwiderstände (z. B. durch naturnahe Gestaltung)
Erschließung zusätzlicher Hochwasserrückhalteräume	Polder Rückhaltebecken (ungesteuert) Rückhaltebecken (gesteuert)
Steigerung der Abflussleistung	Vergrößerung des Fließquerschnitts - Gerinneaufweitungen Erhöhung des Sohlgefälles Verringerung der vorhandenen Fließwiderstände - Glättung Beseitigung lokaler Abflusshindernisse
Gerinneentlastung	Ab- bzw. Überleitung in ein benachbartes Fließgewässer bzw. Einzugsgebiet Umleitungen (Bau von Entlastungsanlagen - z. B. Bypässe, Flutgräben, Entlastungskanäle, Flutmulden) Optimierung der Abflussaufteilung durch Umverteilung der Abflüsse

Anlage E - 2 Hochwasserschutz in der Bauleitplanung

Hochwasserschutzmaßnahme	Beschreibung
Berücksichtigung der HW-Problematik bei der Ausweisung von Baugebieten	Siehe "Gesetzliche Verankerung des HW Schutz in der Bauleitplanung"
Reduzierung der Bodenversiegelung	Siehe "Gesetzliche Verankerung des HW Schutz in der Bauleitplanung"
Maßnahmen zur Regenwasserversickerung	Siehe "Gesetzliche Verankerung des HW Schutz in der Bauleitplanung"
Objektbezogene Anforderungen	Sicherung von Heizöltanks Anwendung bestimmter Gründungsverfahren Anordnung eines Notausgangs bei Mehrfamilienhäusern

Anlage E - 3 Gesetzliche Verankerung des HW Schutz in der Bauleitplanung (Patt und Bechteler, 2001)

Berücksichtigung des Hochwasserschutzes im Flächen-nutzungsplan	Möglichkeit der speziellen Darstellung im Flächennutzungsplan (§ 5 Abs. 2 Nr. 7 BauGB) 1	Flächen, die im Interesse des Hochwasserschutzes und der Regelung des Wasserabflusses frei zuhalten sind, (z. B. Flächen für Deiche, Dämme, Gräben, Hochwasserabflussgebiete)
	Im Flächennutzungsplan sollen gekennzeichnet werden (§ 5 Abs. 3 Nr.1 BauGB)	Flächen, bei denen besondere bauliche Sicherungsmaßnahmen gegen Naturgewalten erforderlich sind
	Nachrichtliche Übernahme in den Flächennutzungsplan (§ 5 Abs. 4 BauGB)	Etwaige Überschwemmungsgebiete nach Wasserrecht
Hochwasserschutz in der Bebauungs-planung	Festsetzung von Flächen im Bebauungsplan (§ 9 Abs. 1 Nr.16 BauGB)	Flächen für Hochwasserschutzanlagen und für die Regelung des Wasserabflusses.
	Festsetzung von Flächen im Bebauungsplan zur Begrenzung der Bodenversiegelung (§ 9 Abs. 1 Nr.10 BauGB; § 9 Abs. 1 Nr.20 BauGB; § 9 Abs. 1 Nr.25a BauGB; § 9 Abs. 1 Nr.25b BauGB)	U. a. Flächen, die von der Bebauung frei zuhalten sind, Festsetzungen von Maßnahmen für Bepflanzungen
	Besondere Kennzeichnung (§ 9 Abs. 5 BauGB)	Flächen, bei denen besondere bauliche Sicherungsmaßnahmen gegen Naturgewalten erforderlich sind.
	Nachrichtliche Übernahme in den Bebauungsplan (§ 9 Abs. 1 Nr.10 BauGB)	Etwaige Überschwemmungsgebiete nach Wasserrecht
Beschränkung der Bodenversiegelung	§ 9 Abs. 1 Nr.10 BauGB; § 9 Abs. 1 Nr.20 BauGB; § 9 Abs. 1 Nr.25a BauGB; § 9 Abs. 1 Nr.25b BauGB; § 40 Abs. 1 Nr.14 BauGB; § 40 Abs. 2 BauGB; § 179 Abs. 1 S.2 BauGB	
Festsetzungen zur Versickerungspflicht von Regenwasser	§ 9 Abs. 1 Nr.20 BauGB; § 9 Abs. 4 BauGB	
Versickerungsmaßnahmen bei einem bauordnungsrechtlichen Versickerungsgebot	§ 42 Abs. 2 HBO; § 56 Abs. 1 Nr.7 Nds. BauO	

Anlage F Systemelemente des Erft-Optimierungssystems

	Original	Typ	Ort	Gewässer
A: HRB Eicherscheid	Einzeleinleiter Erft	Pegel / Einl.	Pegel Schönau	Erft
	Schönau - Eicherscheid			Erft
	Schalter Original oder Variante	Schalter	Eicherscheid	Erft
	Eicherscheid	0-Variante	Eicherscheid	Erft
	HRB Eicherscheid	HWM	Eicherscheid	Erft
SO 1	Schalter Eicherscheid 1 od. 2	Schalter	Eicherscheid Nord	Erft
	Schalter Eicherscheid 1 od. 2	Schalter	Eicherscheid Nord	Erft
	Eicherscheid 0-Variante	0-Variante		Erft
	Mäander Var. 1	Variante 1	Eicherscheid Nord	Erft
	Eicherscheid HRB2 (Variante 2)	Variante 3	Eicherscheid Nord	Erft
	Eicherscheid Var - Münstereifel		Bad Münstereifel	Erft
SO 2	Schalter Original oder Variante	Schalter	Arloff	Erft
	Münstereifel - Arloff	0-Variante		Erft
	Münstereifel - Arloff	Variante 1	Arloff	Erft
	HRB Arloff	Variante 1	Arloff	Erft
	Arloff -Euskirchen			Erft
SO 3	Einzeleinleiter Veybach	Pegel / Einl.	Pegel Burg Veynau	Veybach
	Schalter Original oder Variante	Schalter	Euskirchen West	Veybach
	Schalter Veybach 1 od. 2	Schalter	Euskirchen West	Veybach
	Veybach 0-Variante	0-Variante		Veybach
	Veybach 1 (Gerinne Mod)	Variante 1	Euskirchen West	Veybach
	Überlaufschwelle Veybach 2	Variante 2	Euskirchen West	Veybach
	Veybach 2 (Mäander)	Variante 2	Euskirchen West	Veybach
	Veybach 2 (Wiese)	Variante 2	Euskirchen West	Veybach
	Ablauf Veybach 2	Variante 2	Euskirchen West	Veybach
	Veybach			Veybach
B: HRB Horchheim	Euskirchen - Hausweiler		Euskirchen	Erft
	Pegel Hausweiler	Pegel / Einl.	Pegel Langenich	Erft
	Hausweiler - Hausweiler		Kessenich	Erft
	Schalter Original oder Variante	Schalter	Horchheim	Erft
	Horchheim	0-Variante	Horchheim	Erft
	HRB Horchheim	HWM	Horchheim	Erft
Horchheim - Weilerswist			Erft	
SO 4	Einzeleinleiter Swistbach	Pegel / Einl.	Pegel Langenich	Swistbach
	Swistbach		Weilerswist	Swistbach
	Schalter Original oder Variante	HWM	Swist Weilerswist	Swistbach
	Schalter Original oder Variante	HWM	Erft Weilerswist	Erft
	HRB Bliesheim	Variante 1	Weilerswist	Erft
	Weilerswist - Bliesheim			Erft
Modell 1	Pegel Bliesheim	Pegel / Einl.	Erftstadt	Erft
	Bliesheim - Erftstadt		Erftstadt	Erft

	Abzweig Wiese	HWM	Erfstadt	Erft
	Wiese Erfstadt	HWM	Erfstadt	Erft
C: HRB Niederberg	Pegel Friesheim	Pegel / Einl.	Friesheim	Rotbach
	Niederberg	Schalter	Niederberg	Rotbach
	Niederberg	0-Variante	Niederberg	Rotbach
	HRB Niederberg	HWM	Niederberg	Rotbach
	Rotbach Unterlauf			Rotbach
SO 5	Schalter Original oder Variante	Schalter	Mödrath-Balkhausen	Erft
	Erfstadt - Mödrath 0-Variante	0-Variante	Mödrath-Balkhausen	Erft
	Erfstadt - Mödrath 0-Variante	0-Variante	Mödrath-Balkhausen	Erft
	Erfstadt - Mödrath 1	Variante 1	Mödrath-Balkhausen	Erft
	Erfstadt - Mödrath 1	Variante 1	Mödrath-Balkhausen	Erft
D: HRB Mödrath	Abzweigung TE02	HWM	Mödrath	Erft
	Nat. Retentionsstau	HWM	Mödrath	Erft
	HRB Mödrath	HWM	Mödrath	Erft
	Pegel Mödrath	Pegel / Einl.	Mödrath	Erft
	Mödrath		Mödrath	Erft
SO 6	Einzeleinleiter Neffelbach	Pegel / Einl.	Pegel Langenich	Neffelb.
	Schalter Original oder Variante	Schalter	Langenich	Neffelb.
	Schalter Langenich 1 od. 2	Schalter	Langenich	Neffelb.
	Neffelbach	0-Variante		Neffelb.
	Abzweigung TN42 (Langenich 1)	Variante 1	Langenich	Neffelb.
	Langenich 1 - 0.612Mio	Variante 1	Langenich	Neffelb.
	Abfluss Langenich 1	Variante 1	Langenich	Neffelb.
	Fließweg bei Langenich o. HW	Variante 1	Langenich	Neffelb.
	Abzweigung TN43 (Langenich 2)	Variante 2	Langenich	Neffelb.
	Langenich 2 - 1.56Mio	Variante 2	Langenich	Neffelb.
	Fließweg bei Langenich o. HW	Variante 2	Langenich	Neffelb.
	Abfluss Langenich 2	Variante 2	Langenich	Neffelb.
	Neffelbach			Neffelb.
Modell 2	Mödrath - Kerpen		Kerpen	Erft
	Abzweig Wiese	HWM	Kerpen	Erft
	Wiese bei Kerpen	HWM	Kerpen	Erft
	Kerpen - Bergheim			Erft
	5-S, 6-K, 5-1	Sümpfung	Bergheim-Kerten	Erft
	Pegel Glesch	Pegel / Einl.	Glesch	Erft
	Bündelung Zuflüsse Pos. 10 + 11		Glesch	
	Bergheim-Glesch			Erft
SO 7	Schalter Original oder Variante	Schalter	Garsdorf	
	Schalter Garsdorf 1 oder 2	Schalter	Garsdorf	
	Glesch-Bedburg	0-Variante	Glesch	Erft
	Abzweigung TE01	Variante 1	Garsdorf	
	HRB Garsdorf 900 Ausbau 1	Variante 1	Garsdorf	
	Glesch-Bedburg	Variante 1	Garsdorf	Erft

	Abfluss Garsdorf 900	Variante 1	Garsdorf	
	Abzweigung TE01	Variante 2	Garsdorf	
	HRB Garsdorf 1600 Ausbau 2	Variante 2	Garsdorf	
	Glesch-Bedburg	Variante 2	Garsdorf	Erft
	Abfluss Garsdorf 1600	Variante 2	Garsdorf	
	Abfluss Varianten	Variante 1 + 2	Garsdorf	
Modell	3-S, 4-S	Sümpfung	Bowa Glesch	Erft
	Bedburg-Kaster		Bedburg-Kaster	Erft
	Kaster-Frimmersdorf		Neurath	Erft
SO 8	Schalter Original oder Variante	Schalter	Frimmersdorf	Erft
	Abzweig UE41	Schalter	Frimmersdorf	Erft
	Frimmersdorf-Grevenbroich	0-Variante	Frimmersdorf	Erft
	Wasser in stillgelegtem Tagebau	Variante 1	Frimmersdorf	Erft
	Frimmersdorf-Grevenbroich Var.	Variante 1	Frimmersdorf	Erft
SO 9	Schalter Original oder Variante	Schalter	Wevelinghoven	Erft
	Abzweig TE31	Schalter	Wevelinghoven	Erft
	Grevenbroich-Einm Gillbach	0-Variante		Erft
	Polder Wevelinghoven	Variante 1	Wevelinghoven	Erft
	Grevenbroich-Einm Gillbach Var.	Variante 1	Wevelinghoven	Erft
	Einm. Gillbach - Einm. Norfbach		Neuss	Erft
Modell	Einm. Norfbach - Erftmündung		Neuss	Erft
	Erftm. in den Rhein		Neuss-Grimlingh.	Rhein

Legende
Pegel / Einzeleinleiter
Sümpfungswasser
Hochwasserschutzmaßnahme
Variante
Schalter

Anlage G - 2 Steuerungsdatei mit den Optimierungszielen

* Opt	Bezeichnung	Gruppe	R	Op	Op	Datei	SimGröße	ZielFkt	Evalzeitraum	Referenzwert	IstWert
*			i	Fak					Start	WertTyp	RefWert
*									Ende		
S	VV9_Q_zu	SummeQmax		26		WEL	VV9_1ZU	Diff		MaxWert	0
S	SE17_Q_zu	SummeQmax		23		WEL	SE17_1ZU	Diff		MaxWert	0
S	SVEY_Q_zu	SummeQmax		16		WEL	SVEY_1ZU	Diff		MaxWert	0
S	SE16_Q_ab	SummeQmax		9		WEL	SE16_1AB	Diff		MaxWert	0
S	SE14_Q_ab	SummeQmax		3		WEL	SE14_1AB	Diff		MaxWert	0
S	S132_Q_ab	SummeQmax		1.7		WEL	S132_1AB	Diff		MaxWert	0
S	SRO1_Q_zu	SummeQmax		6		WEL	SRO1_1ZU	Diff		MaxWert	0
S	VV5_Q_zu	SummeQmax		1.5		WEL	VV5_1ZU	Diff		MaxWert	0
S	VN02_Q_zu	SummeQmax		1.5		WEL	VN02_1ZU	Diff		MaxWert	0
S	ST11_Q_zu	SummeQmax		1.1		WEL	ST11_1ZU	Diff		MaxWert	0
S	SE10_Q_zu	SummeQmax		1		WEL	SE10_1ZU	Diff		MaxWert	0
S	SNE2_Q_ab	SummeQmax		16		WEL	SNE2_1AB	Diff		MaxWert	0
S	SE06_Q_ab	SummeQmax		1.1		WEL	SE06_1AB	Diff		MaxWert	0
S	VV2_Q_zu	SummeQmax		1.1		WEL	VV2_1ZU	Diff		MaxWert	0
S	SE01_Q_ab	SummeQmax		1.1		WEL	SE01_1AB	Diff		MaxWert	0
P	Baukosten	Baukosten				WEL	SE01_1AB	Diff		MaxWert	0
* SKos											
* Aggregate											
* Opt	Bezeichnung	Gruppe	R	Op	Op						
				Fak							
P	Baukosten										
* Opt	Bezeichnung	Gruppe	R								
P	SummeQmax	SummeQmax									

Anlage G - 3 Steuerungsdatei mit Optimierungsparametern

*OptimierungsParameter

*=====

*

* Bezeichnung	* Einh.	* Anfangsw.	* Min	* Max
TE04 Volumen	-	1010	202	2020
TV92 Volumen	-	280	56	560
TV08 Volumen	-	400	80	800
TE03 Volumen	-	1376	275	2752
TV05 Volumen	-	796	159	1592
TROT Volumen	-	1160	232	2320
TE02 Volumen	-	1700	340	3400
TN42 Volumen	-	612	122	1224
TN43 Volumen	-	1560	312	3120
TE11 Volumen	-	900	180	1800
TE12 Volumen	-	1600	320	3200
TE41 Volumen	-	1200	240	2400
TE31 Volumen	-	1000	200	2000
TE04 Abgabe	-	2	1	20
TV92 Abgabe	-	10	5	30
TV08 Abgabe	-	10	6	35
TE03 Abgabe	-	30	10	50
TV05 Abgabe	-	10	5	30
TROT Abgabe	-	20	5	40
TE02 Abgabe	-	3	1	15
TN42 Abgabe	-	1	1	10
TN43 Abgabe	-	1	1	10
TE11 Abgabe	-	1	1	10
TE12 Abgabe	-	1	1	10
TE41 Abgabe	-	0.8	0.5	10
TE31 Abgabe	-	0.8	0.5	10

Anlage G - 4 Steuerungsdatei mit Modellparametern

*ModellParameter

*=====

*

*	-----	-----	-----	-----	-----	-----	-----	-----
*	OptParameter	Bezeichnung	Einh.	Datei	Elem./O	Zeile	von	bis
*	<----->	<----->	<----->	<----->	<----->	<----->	<----->	<----->
	TE04 Volumen	TE04 Volumen	-	TAL	TE04	10	24	33
	TV92 Volumen	TV92 Volumen	-	TAL	TV92	11	24	33
	TV08 Volumen	TV08 Volumen	-	TAL	TV08	12	24	33
	TE03 Volumen	TE03 Volumen	-	TAL	TE03	13	24	33
	TV05 Volumen	TV05 Volumen	-	TAL	TV05	14	24	33
	TROT Volumen	TROT Volumen	-	TAL	TROT	15	24	33
	TE02 Volumen	TE02 Volumen	-	TAL	TE02	17	24	33
	TN42 Volumen	TN42 Volumen	-	TAL	TN42	18	24	33
	TN43 Volumen	TN43 Volumen	-	TAL	TN43	19	24	33
	TE11 Volumen	TE11 Volumen	-	TAL	TE11	20	24	33
	TE12 Volumen	TE12 Volumen	-	TAL	TE12	21	24	33
	TE41 Volumen	TE41 Volumen	-	TAL	TE41	22	24	33
	TE31 Volumen	TE31 Volumen	-	TAL	TE31	23	24	33
	TE04 Abgabe	TE04 Abgabe 1	-	FKT	TE04	61	66	68
	TE04 Abgabe	TE04 Abgabe 2	-	FKT	TE04	62	66	68
	TE04 Abgabe	TE04 Abgabe 2	-	FKT	TE04	63	66	68
	TV92 Abgabe	TV92 Abgabe 1	-	FKT	TV92	77	66	68
	TV92 Abgabe	TV92 Abgabe 2	-	FKT	TV92	78	66	68
	TV92 Abgabe	TV92 Abgabe 2	-	FKT	TV92	79	66	68
	TV08 Abgabe	TV08 Abgabe 1	-	FKT	TV08	88	66	68
	TV08 Abgabe	TV08 Abgabe 2	-	FKT	TV08	89	66	68
	TV08 Abgabe	TV08 Abgabe 3	-	FKT	TV08	90	66	68
	TE03 Abgabe	TE03 Abgabe 1	-	FKT	TE03	97	66	68
	TE03 Abgabe	TE03 Abgabe 2	-	FKT	TE03	98	66	68
	TV05 Abgabe	TV05 Abgabe 1	-	FKT	TV05	107	66	68
	TV05 Abgabe	TV05 Abgabe 2	-	FKT	TV05	108	66	68
	TV05 Abgabe	TV05 Abgabe 3	-	FKT	TV05	109	66	68
	TROT Abgabe	TROT Abgabe 1	-	FKT	TROT	142	66	68
	TROT Abgabe	TROT Abgabe 2	-	FKT	TROT	143	66	68
	TROT Abgabe	TROT Abgabe 3	-	FKT	TROT	144	66	68
	TROT Abgabe	TROT Abgabe 4	-	FKT	TROT	145	66	68
	TE02 Abgabe	TE02 Abgabe 1	-	FKT	TE02	162	66	68
	TE02 Abgabe	TE02 Abgabe 2	-	FKT	TE02	163	66	68
	TE02 Abgabe	TE02 Abgabe 3	-	FKT	TE02	164	66	68
	TE02 Abgabe	TE02 Abgabe 4	-	FKT	TE02	165	66	68
	TN42 Abgabe	TN42 Abgabe 1	-	FKT	TN42	172	66	68
	TN42 Abgabe	TN42 Abgabe 2	-	FKT	TN42	173	66	68
	TN43 Abgabe	TN42 Abgabe 1	-	FKT	TN43	180	66	68
	TN43 Abgabe	TN42 Abgabe 2	-	FKT	TN43	181	66	68
	TE11 Abgabe	TN42 Abgabe 1	-	FKT	TE11	188	66	68
	TE11 Abgabe	TN42 Abgabe 2	-	FKT	TE11	189	66	68
	TE12 Abgabe	TN42 Abgabe 1	-	FKT	TE12	196	66	68
	TE12 Abgabe	TN42 Abgabe 2	-	FKT	TE12	197	66	68
	TE41 Abgabe	TN42 Abgabe 1	-	FKT	TE41	204	66	68
	TE41 Abgabe	TN42 Abgabe 2	-	FKT	TE41	205	66	68
	TE31 Abgabe	TN42 Abgabe 1	-	FKT	TE31	212	66	68
	TE31 Abgabe	TN42 Abgabe 2	-	FKT	TE31	213	66	68
*	-----	-----	-----	-----	-----	-----	-----	-----

Anlage H Einstellungen zur Kontrolle der Algorithmen

Da beim memetischen Algorithmus auf bereits vorhandene Operatoren zurückgegriffen sind dementsprechend bei jeder Optimierung auch deren Steuerparameter einzustellen und die Operatoren auszuwählen. Die Operatoren des nominell diskreten Algorithmus wurden detailliert im Kapitel 3.1.4 erprobt und in Kapitel 3.1.5 diskutiert. Die Steuerparameter und Operatoren für den reellwertigen Algorithmus wurden detailliert in den Arbeiten Huebner (2004) und Muschalla (2006) diskutiert und sind der Vollständigkeit halber in den unten (Anlage H - 1, Anlage H - 2 und Anlage H - 3) aufgelistet. Für den memetischen Algorithmus sind neben den zusätzlichen Parametern für die Abstimmung zwischen den Operatoren sind die klassischen evolutionären Steuerparameter erforderlich:

- **Startpopulation:** Gibt an ob die Startpopulation aus zufälligen oder vorab festgelegten Parametern besteht.
- **Optimierungsmodus:** Gibt an ob die sequenzielle Prozedur oder die Prozedur mit selektierten reellen Parametern verwendet werden soll.
- **Anzahl der Generationen g :** Gibt die Zahl der Generationen an.
- **Anzahl der Eltern μ und Anzahl der Nachfahren λ** bilden den Selektionsdruck μ/λ .
- **Nominell Diskreter Reproduktionsoperator:** Definiert den zu verwendenden nominell diskreten Reproduktionsoperator.
- **Reeller Reproduktionsoperator:** Definiert den zu verwendenden reellen Reproduktionsoperator.
- **Nominell Diskreter Mutationsoperator:** Definiert den zu verwendenden nominell diskreten Mutationsoperator.
- **Reeller Mutationsoperator:** Definiert den zu verwendenden reellen Mutationsoperator.
- **Anzahl der Schnittpunkte:** Legt die Anzahl der Schnittpunkte für einen nominell diskreten Reproduktionsoperator
- **Mutationsrate:** Gibt die Mutationsrate für den nominell diskreten Mutationsoperator an.
- **Memory Größe:** Legt die Größe des Memories für den \mathcal{E}_{Mem} Selektionsoperator fest.

-
- **Memory Zugriff:** Legt fest auf welche Eltern für den reellwertigen Teil zurückgegriffen wird.

Anlage H-1 Steuerparameter des kombinatorischen Algorithmus

```
Private _Startparameter As EVO_STARTPARAMETER
Private _OptModus As String
Private _n_Generations As Integer
Private _n_Parents As Integer
Private _n_Children As Integer
Private _OptStrategie As EVO_STRATEGIE
Private _OptReprodOp As CES_REPRODOP
Private _k_Value As Integer
Private _OptMutOperator As CES_MUTATION
Private _pr_MutRate As Integer
Private _Is_SecPop As Boolean
Private _Is_SecPopRestriction As Boolean
Private _n_MemberSecondPop As Integer
Private _n_Interact As Integer
```

Anlage H-2 Steuerparameter des Memetischen Evolutionären Algorithmus

```
Private _RealOptEnabled As Boolean
Private _is_RealOpt As Boolean
Private _HybridType As HYBRID_TYPE
Private _Mem_Strategy As MEMORY_STRATEGY
Private _n_PES_MemSize As Integer
Private _is_PES_SecPop As Boolean
Private _n_PES_MemSecPop As Integer
Private _n_PES_Interact As Integer
Private _is_PopMutStart As Boolean
```

Anlage H-3 Steuerparameter des reellwertigen Algorithmus

```
Private _OptModus As EVO_MODUS
Private _Strategie As EVO_STRATEGIE
Private _Startparameter As EVO_STARTPARAMETER
Private _N_Gen As Integer
Private _N_Eltern As Integer
Private _N_Nachf As Integer
Private _Reproduktionsoperator As PES_REPRODOP
Private _Mutationsoperator As PES_MUTATIONSOP
Private _N_RekombXY As Integer
Private _Is_DiversityTournament As Boolean
Private _DnStart As Double
Private _DnEpsilon As Double
Private _IsDnVektor As Boolean
Private _DnC As Double
Private _N_Interact As Integer
Private _Is_Begrenzung As Boolean
Private _N_MaxMembers As Integer
Private _Is_POPUL As Boolean
Private _N_Runden As Integer
Private _N_Popul As Integer
Private _N_PopEltern As Integer
Private _PopEltern As EVO_POP_ELTERN
Private _PopStrategie As EVO_STRATEGIE
Private _PopPenalty As EVO_POP_PENALTY
```

Curriculum Vitae

Education and training

Dates	2004 – 2010
Title of qualification aimed at	University degree: Doctor of Philosophy (Doktor-Ingenieur)
Principal occupational skills covered	Engineering Hydrology and Water Management
Name and address of organisation	Darmstadt University of Technology, Karolinenplatz 5, 64289 Darmstadt, Germany
Level in international classification	ISCED 6

Dates	1996 – 2004
Title of qualification awarded	University degree: Master of Science (Diplom-Ingenieur)
Principal subjects/ occupational skills covered	Engineering Hydrology and Water Management, Comput- er Science in Civil Engineering, Steel Construction
Name and address of organisation	Darmstadt University of Technology, Karolinenplatz 5, 64289 Darmstadt, Germany
Level in international classification	ISCED 5A

Dates	1987- 1995
Title of qualification awarded	Abitur
Principal subjects skills covered	Mathematics, physics, german
Name and address of organisation	Max-Planck-Gymnasium, Realschulstraße 9, 64823 Groß- Umstadt, Germany
Level in international classification	ISCED 3A

Committees

Dates	2006-2008 – onwards
Occupation or position held	Member
Scientific topics of the committee	Decision support systems in river basin management
Name and address of the committee	DWA (German Association of Water Management, Waste Water and Waste Management)
Type of business or sector	Hydrology and water management

