Techniques for Efficient Interactive Configuration of Distribution Networks

Tarik Hadžić and **Andrzej Wąsowski** and **Henrik R. Andersen**
Computational Logic and Algorithms Group, IT University of Copenhagen,
Rued Langgaards Vej 7, DK-2300 Copenhagen S, Denmark
{tarik,wasowski,hra}@itu.dk

Abstract

Recovering from power outages is an essential task in distribution of electricity. Our industrial partner postulates that the recovery should be interactive rather than automatic: supporting the operator by preventing choices that destabilize the network.

Interactive configurators, successfully used in specifying products and services, support users in selecting logically constrained parameters in a sound, complete and backtrack-free manner. Interactive restoration algorithms based on reduced ordered binary decision diagrams (BDDs) had been developed also for power distribution networks, however they did not scale to the large instances, as BDDs representing these could not be compiled.

We discuss the theoretical hardness of the interactive configuration and then provide techniques used to compile two classes of networks. We handle the largest industrial instances available. Our techniques rely on symbolic reachability computation, early variable quantification, domain specific ordering heuristics and conjunctive decomposition.

1 Introduction

Interactive configuration is an application of constraint satisfaction (CSP) that assists a user in her search for a valid variable assignment (a *configuration*) in a combinatorial problem. Its major application areas include sales of services (airplane tickets, insurance policies) and goods (personal computers, cars).

The assistance takes the form of proposing choices that are globally consistent, i.e. always lead to a legal solution to the problem. In CSP terms interactive configurator enforces *generalized arc consistency* (GAC) wrt. implicit conjunction of all constraints, i.e. it computes a *valid domain* for each unassigned variable, consisting of all valuations that are guaranteed to be globally completable. As a consequence, as long as a user assigns values from valid domains the interaction is *complete* (all valid configurations are reachable through user interaction) and *backtrack-free* (a user is never forced to change an earlier choice). Additionally the computation must run in *real-time*, to facilitate use in interactive setting.

For general CSP models over finite domains, enforcing GAC is NP-hard. Therefore, in order to provide real-time guarantees, current approaches use off-line compilation of the CSP model into a tractable data structure representing the space of all valid configurations [Møller *et al.*, 2001; Amilhastre *et al.*, 2002]. It has been observed that the compiled representations based on reduced ordered binary decision diagrams (BDD) [Bryant, 1986], although having exponential worst-case size, can usually be kept small for the industrial instances of product configuration [Subbarayan *et al.*, 2004].

Here we advance a state of the art for a recent application of interactive configuration in the domain of power supply restoration [T. Hadzic and H. R. Andersen, 2005]. The PSR domain has received a lot of attention, after several blackouts had caused serious financial loss and had raised security considerations throughout North America and Europe. These in turn have inspired a host of research on automatic recovery from power outages based on search and planning techniques [Thiébaux and Cordier, 2001; Bertoli *et al.*, 2002] first for the high voltage transport networks, and later for the more complex and dense distribution networks.

Our industrial partner NESA A/S, a power distribution operator in the Copenhagen area, insisted that the power restoration process should be interactive rather than automatic: leaving the control to the operator interactively reconfiguring the network, while still guiding her in this process. This makes the standard solutions based on network-flow algorithms inadequate since instead of finding just one (possibly optimal wrt. some cost) configuration we need to reason about all possible valid network configurations. In addition we were strongly required to take into account the entire combinatorial hardness of the problem, guaranteeing not only uninterrupted flow of electricity in circuits (*connectivity constraints*), but also meeting constraints on the maximum load carried by a line (*load constraints*).

Due to numerous cyclic dependencies, the industrial instances provided by NESA, proved to be much harder than any of the product configuration instances we had encountered so far. They were also much larger than PSR instances seen elsewhere, making our existing configurators unable to work with them. Basic BDD based techniques (see for example the *monolithic approach* in [T. Hadzic and H. R. Andersen, 2005]) required representing the network with a single BDD. We investigated a number of techniques to increase the size of the network that can be compiled. In this paper we present the most successful of these. We describe improvements in the PSR models achieved by reducing the number of network elements represented in a BDD, and use of PSR specific variable orderings. We also introduce an alternative way to compile BDDs representing connectivity constraints only—an important subclass of the problem, widely used in other PSR research. Finally, we describe our decomposition approach that allows us to scale to the largest instance in our collection (3 power sources, 119 consumer sinks, 146 line segments, see Figure 2). We believe that this is the largest network that can be handled in a backtrack-free and complete manner, under given time requirements so far. Our instances are publicly available at `http://www.itu.dk/~tarik/psr`.

Related work BDDs have been successfully applied in product configuration [Subbarayan *et al.*, 2004]. However, the network topology of PSR is much harder for BDDs to represent (standard compilation techniques experience size explosion of BDDs even for smaller instances). Interaction over decomposed set of BDDs has been explored before. In [Meer *et al.*, 2006] the authors provide strong guarantees for interaction over acyclic network of BDDs that can be dynamically extended by unlimited number of new BDDs. In contrast, we do not provide support for dynamic structures of unbounded size, but we handle cyclic dependencies.

The PSR problem has been investigated in the context of automated planing. However, the instances presented were relatively small [Thiébaux and Cordier, 2001] and disregarded the combinatorial hardness of the problem caused by the load constraints. Recently it has been shown that the PSR planning problem is easy when these constraints are ignored [Helmert, 2006]. We show that the interactive configuration of PSR, as postulated by NESA, is NP-hard under load constraints and polynomial when these constraints are ignored.

Our work is likely to be of interest for other applications exploiting BDD representations of networks such as automatic reconfiguration [T. Hadzic and H. R. Andersen, 2005], automated planning [Jensen *et al.*, 2004] and reliability analysis [Dutuit *et al.*, 1997]. The techniques should perform well also for other kinds of networks (eg. distributing water, natural gas, sewage, or for telecommunication networks).

We proceed as follows. Section 2 describes the problem defining load networks and connectivity networks, their complexity properties and the basics of our models. Section 3 discusses techniques supporting compilation of load networks up to the medium size instances. Section 4 describes techniques for compiling connectivity networks up to the largest instances available, while Section 5 brings a decomposition approach that allows interactive configuration of the largest available instances of load networks. Experimental evaluation is discussed throughout sections 3–5. We summarize and conclude in Section 6.

2 PSR Configuration Problems

We view a power network as a directed graph $G(V,E)$, where vertices represent power *sources* P (supplying electrical current) and *sinks* S (consuming it): $P \subseteq V, S = V \setminus P$. In reality sinks are transformer stations that transmit electricity to final consumers. Each edge $e = (v_1, v_2) \in V \times V$, represents a power *line* capable of transporting current between v_1 and v_2, in any direction. The state of e is *forward* if e is transmitting a non-zero current from v_1 to v_2, and *backward* when transmitting from v_2 to v_1. Otherwise the state is *off*. We denote the current load on e as $l_e \geq 0$. When e is off then $l_e = 0$.

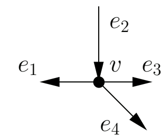

Figure 1: Lines e_1, e_2, e_3, e_4 incident with sink v. For simplicity arrows denote the direction of flow, not the edge directions. Only one of the lines can lead the power into v (connectivity rules), e_2 here. If v is on then according to Kirchhoff's law: $l_{e_2} = |v| + l_{e_1} + l_{e_3} + l_{e_4}$

A sink v consumes up to $|v|$ units of power. A sink can be *on* iff it is connected to a line in the forward or backward state that provides at least $|v|$ units of current. Otherwise a sink is *off*. A sink can be off even when connected to a powered line. In such case current flows through it without any loss.

The following *connectivity constraints* must always hold:

- No short circuits: undirected cycles with sources but without active sinks (resistance) are forbidden.
- No self feeding cycles: undirected cycles that have no active sinks and no sources are forbidden.
- No sink is powered from more than one power line.

Most of the distribution network instances that we have seen in the PSR research enforce only the above connectivity rules, effectively ignoring the relation between the consumption of sinks and the capacity of lines. This means that a sink can be switched *on* based only on whether it can be connected to a powered line. We call these special subclass of the networks the *Connectivity Distribution Networks*. They are important to study in themselves, as being the basis for most of the PSR research they can serve as a reference in comparisons. However, our industrial partner also requires the following *load distribution constraints* to hold:

- Kirchhoff's laws for load distribution must be preserved as illustrated in Figure 1.
- For each line e its current load l_e must not exceed a constant maximum capacity |e|: $l_e \leq |e|$.

In case of a line failure, an operator should be supported to interactively configure the affected part of the network. The advantage of interactive instead of automatic recovery lies in the fact that since a lot of information about the network cannot be encoded in CSP models, an operator is likely to make more qualified decisions about how to reconfigure than an automatic system. [T. Hadzic and H. R. Andersen, 2005] describe this interactive reconfiguration process in detail.

The operator interactively decides which power lines to use, and which to keep *off*. The broken lines are forced to be *off*, and some important sinks (hospitals, etc) should be always *on*. In each interaction step, after fixing a line or a sink, the operator gets valid domains for remaining network elements. We distinguish the problem of computing CONNECTIVITY-VALID-DOMAINS from LOAD-VALID-DOMAINS, depending on which of the two kinds of distribution networks we are working with.

Problem (CONNECTIVITY-VALID-DOMAINS). *For a set of lines $E_{off} \subseteq E$ required to be off and the set of sinks $S_{on} \subseteq S$ required to be on decide for each line $e \in E \setminus E_{off}$ whether it is possible to power all sinks in S_{on} only using lines in $E \setminus (E_{off} \cup \{e\})$, and for each sink $v \in S \setminus S_{on}$ whether the set $S_{on} \cup \{v\}$ can be powered only using lines in $E \setminus E_{off}$, while connectivity constraints are satisfied.*

Theorem 1. CONNECTIVITY-VALID-DOMAINS *is easy.*

Proof sketch. The following algorithm checks in $O(mn)$ time if a given selection of E_{off} and S_{on} can be satisfied (m is the number of lines and n is the number of sinks):

CONNECTIVITY-SAT$(G, E_{\text{off}}, S_{\text{on}})$

$G' \leftarrow G(V, E \setminus E_{\text{off}})$, $S'_{\text{on}} \leftarrow \emptyset$
for each power source p in P
    do DFS traversal of nodes in G' reachable from p
        Add to S'_{on} all visited nodes
return $(S_{\text{on}} \subseteq S'_{\text{on}})$

It suffices to call CONNECTIVITY-SAT $|E \setminus E_{\text{off}}| + |S \setminus S_{\text{on}}|$ times to verify if any of the remaining lines can be forced off, or if any of the remaining sinks can be forced on. □

Similar models of *connectivity* networks are used as benchmarks for planing under uncertainty [Thiébaux and Cordier, 2001]. A complexity analysis of [Helmert, 2006] indicates that these networks are easy also for planning algorithms.

Problem (LOAD-VALID-DOMAINS). *Given a set of lines $E_{off} \subseteq E$ required to be off and the set of sinks $S_{on} \subseteq S$ required to be on decide for each line $e \in E \setminus E_{off}$ whether it is possible to power all sinks in S_{on} only using lines in $E \setminus (E_{off} \cup \{e\})$, and for each sink $v \in S \setminus S_{on}$ whether the set $S_{on} \cup \{v\}$ can be powered only using lines in $E \setminus E_{off}$, while both the connectivity and load constraints are satisfied.*

Let us now formulate an auxiliary problem:

Problem (SET-SUM-PARTITION). *Given a finite set S of positive integers and two integer constants c_1 and c_2 such that $\sum_{s \in S} s \leq c_1 + c_2$ decide whether there exist subsets S_1 and S_2 such that $S = S_1 \uplus S_2$ and for $i = 1, 2$. $\sum_{s \in S_i} s \leq C_i$.*

The above problem is NP-hard [T. Hadzic and H. R. Andersen, 2006], which can be shown by a straightforward reduction from SUBSET-SUM [Garey and Johnson, 1979, p.223].

Theorem 2. LOAD-VALID-DOMAINS *involves solving NP-hard problems.*

Proof sketch. We show that decision problems contained in LOAD-VALID-DOMAINS are NP-hard, by reduction from SET-SUM-PARTITION. For an instance S, c_1, c_2 create a network as follows: for each $k \in S$ create a sink v_k with consumption $|v_k| = k$. Add one source p connected to new dummy sinks s_1, s_2, s_3 such that $|s_1| = |s_2| = |s_3| = 1$. Let $|e(p, s_1)| = c_1$, $|e(p, s_2)| = c_2$ and $|e(p, s_3)| = 1$. For each sink v_k add lines $e(s_1, v_k)$ and $e(s_2, v_k)$ with capacity $\max(c_1, c_2)$. Notice that due to connectivity constraints, in any legal configuration each v_i sink can be powered from a line coming from either s_1 or s_2, but never from both. Consider solving LOAD-VALID-DOMAINS on the created graph with $E_{\text{off}} = \emptyset$ and $V_{\text{on}} = \{v_k \mid k \in S\}$. This requires deciding whether s_3 can be on, while the other constraints are satisfied. If it cannot, then there is no suitable partition of S. If it can then the answer to the original problem is yes (to obtain a witness put into S_1 all $k \in S$ such that v_k is powered via s_1 and into S_2 all k powered via s_2). □

Since the cost of computing valid domains for spaces represented in BDDs is given by a polynomial parameterized by the BDD size [T. Hadzic *et al.*, 2006], Thm. 2 implies that there exist instances of load networks with exponentially large BDD representations, regardless of the variable ordering. Additionally the two complexity results above may be perceived as indications (not proofs!) how hard it is to represent the respective problems using BDDs. Indeed we will soon experience that it is relatively easier to construct representations for connectivity networks than for load networks.

We have built models of NESA's instances using a customized version of a BDD-based configuration library CLab [Jensen, online]. NESA require to model their network as a *load distribution network*. The part of the network under consideration contains lines of maximum current capacity of 260 A, and transformer stations of size 400 KVA - which corresponds to current consumption of about 22 A for each transformer. This means that a power line with maximum current intensity can feed at most 13 sinks. This allows us to discretize the domain of current load from $[0, 260]$ to $\{0, 1, \ldots 13\}$, assuming that each transformer station consumes one unit of current. It suffices to enforce Kirchoff's distribution laws over these discrete domains to achieve a close and sound approximation of network models with continuous current values: for every configuration satisfying our discrete model there is a range of continuous valuations satisfying the real-valued model.

For each vertex $v \in V$ we have a variable $v_{\text{pow}} \in \{\text{on}, \text{off}\}$ indicating whether it consumes (produces) current or whether it is idle. For each line $e \in E$ we introduce a variable indicating the direction of the current flow $e_{\text{dir}} \in \{\text{off}, \text{backward}, \text{forward}\}$, and a variable modelling the load $l_e \in \{0, 1, \ldots 13\}$. Kirchoff's laws are modelled using finite arithmetic constraints implicitly expanded by CLab into boolean formulæ.

3 Compilation Techniques for Load Networks

The theoretical intuition that BDDs representing load networks are hard to construct, has been confirmed in practice for our instances. Structural properties of these instances are reported in Table 1 (the instances have been created by Henney, Bak, Jensen and Sonne [T. Bak and S. Henney, 2004; Lars Sonne and Rene Jensen, 2005] in collaboration with NESA). The first column in the table shows names of the instances. The subsequent three columns list the numbers

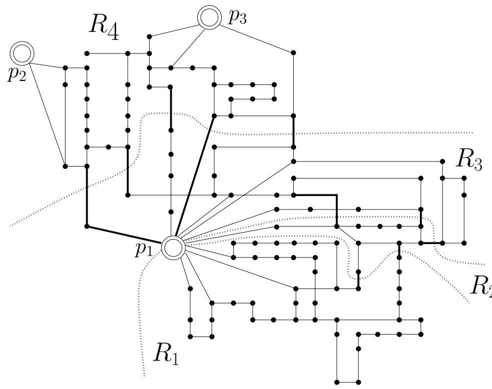

Figure 2: *Complex*: the largest instance.

of lines, sinks and power sources respectively. The size of resulting BDD is indicated in the fifth column. Even the medium sized instances lead to enormous BDDs, and the largest ones cannot be compiled.

The largest instance, *Complex*, has been set by NESA as a goal for the first stage of our collaboration. We were focused on scaling our techniques to be able to handle it. Figure 2 shows the topology of this instance. We believe that *Complex* is the largest real-world instance publicly available.

We have begun with an observation that for an operator configuring the network, the information about loads is not necessary, as long as the load constraints are guaranteed to be met. It suffices to just represent the *on/off* state of sinks and lines. The BDD can be decreased by projecting only on relevant non-load variables. Any satisfying configuration of these variables in the projected BDD can be extended to a configuration including loads in the original BDD.

To benefit from the projection already *during* compilation we use *early variable quantification* [Meinel and Theobald, 1998, p. 195]. CLab compiles models by compiling the constraints separately and then conjoining the results. Early variable quantification allows to quantify out variables at intermediate steps of the long conjunction. As soon as a load variable is not appearing in constraints remaining to be conjoined, it can be existentially quantified. The technique keeps the intermediary BDDs smaller, making it easier to reach the final result (it is known that the final BDD is often much smaller than the biggest intermediate BDDs in a long conjunction).

Table 1: Benchmark properties.

instance name	# of lines	# of sinks	# of sources	simple size(kb)	quantified size (kb)
Std-diagram	13	7	2	215	7.3
1-6+22-32	21	17	2	455	31.6
Complex-P2	24	18	1	3 282	252
Complex-P3	28	19	1	61 240	418
1-32	38	32	2	448 360	4 154
Large	66	56	2	-	-
Complex	146	119	3	-	-

Despite employing several quantification scheduling heuristics, we were unable to compile *Large* and *Complex* (see the last column in Table 1). Nevertheless we have observed much smaller intermediary BDDs, hinting that this technique may be helpful in combination with other improvements. More importantly we have observed a drastic decrease of the final BDD sizes (due to projection). Such a big decrease is not surprising in fact, given that more than half of all Boolean variables in the model were only encoding loads.

It is well known that the size of BDD representations of logical rules strongly depends on the *variable ordering*. The orderings for BDDs in Table 1 were created using a general purpose *Fan-In* heuristic [Meinel and Theobald, 1998, p.146–147] that operates directly on logical rules. Since our model follows the network topology we decided to apply heuristics operating directly on the graph structure instead. We recognized a direct relationship between the topological distance of network elements (sinks, lines) and the logical interdependencies between the variables encoding them. Because of that we place each vertex variable v_{pow} in proximity to variables e_{dir}, l_e for all the lines incident to v that were not already placed with another vertex. Then, in order to place vertex groups with respect to each other, we search for an optimal *linear layout* of the input graph $G(V, E)$ [Diaz *et al.*, 2002] using the *Minimum Linear Arrangement* (MINLA) measure as the objective cost. Other measures might be well suited, too.

After constructing and applying this ordering we observed a significant improvement in the size of the generated BDDs. We finally managed to compile *Large* to: 8 483kb (1 811kb after quantifying out loads). Nevertheless, the combination of these two techniques still does not suffice to reach our milestone: interactive configuration of *Complex*.

4 Techniques for Connectivity Networks

We should now present a compilation technique for *Connectivity Distribution Networks*. We have shown in Section 2 that there exists a truly polynomial algorithm for CONNECTIVITY-VALID-DOMAINS not using BDDs. Still, the BDDs for connectivity networks are interesting in themselves. Besides being useful for comparison with other PSR research, they can be used for cost bounded interactive configuration [Hadzic and Andersen, 2006], where an operator can interactively prune valid domains based on some notion of cost. For example, an operator could limit the maximum number of not-powered sinks, or if there is a cost assigned to changing the status of each network line, then the operator could limit the total cost of introduced changes.

After experiencing very large intermediate BDDs when conjoining connectivity constraints in regular fashion we have developed an alternative way to compile connectivity networks. We remodelled the problem as a *transition system*, such that its reachable state space equals the set of all legal configurations. The construction is rather simple: in the initial state all sinks are *off*, and every transition step only powers lines and sinks in ways that do not violate connectivity constraints (lines cannot close cycles, powered sinks/lines should be connected to powered lines). We compute the state space by using the regular algorithm, which applies the transition relation transitively to the initial state until a fixpoint is reached [Meinel and Theobald, 1998, p.185].

We have created the instances of connectivity networks from the instances of Section 3 by imposing connectivity constraints instead of load constraints and maintaining the topology. We have experienced much smaller intermediate BDDs than when using the regular compilation scheme of CLab. Also we were able to compile BDDs for all the instances, including *Large* (52.5 KB or approx. 2 191 nodes) and *Complex* (1961 KB and 83 572 nodes).

5 Decomposition for Load Networks

We have tried a range of other techniques to scale up to our biggest instance and only the *conjunctive decomposition* was successful. It is strongly related to [Meer *et al.*, 2006]. We divide the graph $G(V,E)$ into overlapping components $G_1(V_1,E_1),\ldots,G_n(V_n,E_n)$, $V_i \subseteq V$, $E_i \subseteq E$ such that every vertex belongs to exactly one component and each edge belongs to at least one and at most two components. Cross component edges belong to two components: $\forall e(v_1,v_2).\, v_1 \in V_i \wedge v_2 \in V_j \wedge i \neq j \Rightarrow e \in E_i \bigcap E_j$. Remaining edges belong to single components. The sink consumption, and load capacities of lines are naturally inherited from the global problem. We refer to the shared lines between regions as *interfaces*: $I_{ij} = E_i \bigcap E_j$.

We compile a BDD ϕ_i for each component G_i separately. Since our modelling closely follows the network topology, only minor modifications are needed to generate the logical rules representing the subnetworks (in particular the load levels and directions on interface lines should not be quantified).

Computing valid domains for conjunctive partitions is NP-hard in general, even if represented by BDDs. However, valid-domains can be calculated in the time polynomial in the sum of the sizes of components, if all the component BDDs ϕ_i are *globally consistent* [Dechter, 2003, p. 67], i.e. if for each component $i = 1 \ldots n$ and for every satisfiable assignment σ_i to ϕ_i there exist a satisfiable assignment σ to $\phi = \phi_1 \wedge \cdots \wedge \phi_n$ which is an extension of σ_i (for acyclic decompositions this reduces to *arc consistency*). The computation of valid domains over the globally consistent space ϕ reduces to standard BDD-based computation over each subnetwork separately: $\textsc{Load-Valid-Domains}(\phi) = \bigcup_{i=1}^{n} \textsc{Load-Valid-Domains}(\phi_i)$, giving an algorithm of cost polynomial in the size of the component BDDs.

So once a conjunctive decomposition is constructed the main problem lies in maintaining global consistency of the network throughout the configuration process. One successful way to do [Meer *et al.*, 2006]. The authors observe that the growth of BDDs at runtime can be exponential in the size of the interface sets I_{ij}, making it essential to find a partitioning with small interfaces. Observe that the use of conjunctive decomposition sacrifices the polynomial response time guarantees of the previous sections. The sizes of the BDDs created during interaction may be exponential, but only exponential in a small number (the interface size). The experiments show that the worst-case time complexity does not appear in practice.

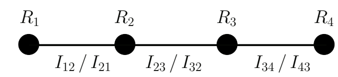

Figure 3: Decomposing *Complex* to regions R_1, R_2, R_3, R_4

After a user assigns $x \leftarrow v$, the solution space ϕ_i to which x belongs is restricted by assigning $\phi_i \leftarrow \phi_i \wedge x = v$. This may make some of the neighboring components allow inconsistent assignments. Therefore a synchronization step is needed, where a global consistency is restored. The new solution space of ϕ_i is projected on all of its neighboring interfaces I_{ij} $((\phi_i \wedge x = v)\!\downarrow\! I_{ij})$ and sent towards the respective neighbors ϕ_j. Then the neighbors propagate the information recursively until a fixpoint is reached. The main algorithm, stripped of technicalities, is presented below (assume that the node i has already been restricted by $x = v$):

$\textsc{Restore-Consistency}(i, \phi_1, \ldots, \phi_n)$

$S \leftarrow \{i\}$
while S is not empty
    do remove some element n from S
        for each interface I_{nm} incident with component n
            do $\phi'_m \leftarrow \phi_m$
                $\phi_m \leftarrow (\phi_n\!\downarrow\! I_{nm}) \wedge \phi_m$
                if $\phi'_m \neq \phi_m$ **then** add m to S

We have partitioned *Complex* into four geographical components R_1–R_4 as shown in Figures 2–3. Each region shares with its neighbors line variables l_e, e_{dir} from the interface sets I_{ij}. The interface sizes are 12, 18 and 30 bits in this example. The BDDs for each of the components are built separately, using the techniques described in previous sections. The variable ordering is optimized for each of the modules separately, using the MINLA heuristic. This means that on top of the projections, we also need to perform renamings when restoring the consistency. In return we benefit from being able to optimize each ordering separately. The sizes of the component BDDs are 403 138 nodes (9 769kb) for R_1, 169 585 (4 497kb) for R_2, 136 636 (3 578kb) for R_3, and 126 976 nodes (3 240kb) for R_4. These BDDs could be made smaller in an industrial quality implementation by introducing more components. Still, the numbers were satisfactory for our „proof of concept" implementation.

The computation of valid domains for *Complex* lasts less than 0.5s (using a state of the art implementation from CLab). The restoration of global consistency takes less than 3s. using our crude implementation. Because the structure is so simple, the restoration never requires more than four conjunctions, but our simplistic implementation often does twice as many. The experiments have been carried on a 3.2Ghz PC. The times decrease dramatically if the network is nearly configured, which is the case in most of the realistic situations.

6 Conclusions and Future Work

We have discussed a range of techniques for interactive configuration of power distribution networks. We have shown that the interactive configuration over *Connectivity Distribution Networks* is easy, while our more realistic model of *Load Distribution Networks* makes interaction NP-hard.

We have shown and experimentally evaluated a range of techniques useful in compilation, interactive configuration, and reconfiguration of solution spaces representing power supply networks. We have described a successful attempt to model the problem in propositional logics and finite domain arithmetics with judicious use of hiding variables and early quantification scheduling. We have proposed a bottom-up way of constructing the solution space as a result of reachable state space computation with a custom transition relation. Finally, by application of conjunctive decomposition, we were able to efficiently interact with the largest instance obtained from our industrial partner: a highly cyclic network of 3 power sources, 119 nodes and 146 line segments.

In future we will try to scale our approach to even bigger instances, and evaluate it on topologies that cannot be easily decomposed into acyclic component graphs (we already support cyclic decompositions, but we have not evaluated the efficiency on such cases, yet). We intend to improve the implementation of our configurator over decomposed networks, and develop automatic heuristics for decomposition.

We would like to quantify theoretically the dependency between the cyclicity of instances and their BDD representation sizes, and to investigate representations other than BDDs.

References

[Amilhastre *et al.*, 2002] J. Amilhastre, H. Fargier, and P. Marquis. Consistency restoration and explanations in dynamic CSPs-application to configuration. *Artificial Intelligence*, 135(1-2):199–234, 2002.

[Lars Sonne and Rene Jensen, 2005] Lars Sonne and Rene Jensen. Power Supply Restoration. Master's thesis, Department of Innovation, IT University of Copenhagen, 2005.

[T. Bak and S. Henney, 2004] T. Bak and S. Henney. Power Supply Restoration—a Constraint-based Model for Reconfiguration of 10KV Electrical Distribution Networks. Student term report. IT University of Copenhagen, 2004.

[T. Hadzic and H. R. Andersen, 2005] T. Hadzic and H. R. Andersen. Interactive Reconfiguration in Power Supply Restoration. In *11th International Conference on Principles and Practice of Constraint Programming CP'05, Sitges, Spain*, 2005.

[T. Hadzic and H. R. Andersen, 2006] T. Hadzic and H. R. Andersen. Combining BDDs with Cost-Bounding Constraints for Interactive Configuration. In *Doctoral Programme Proceedings of CP'06*, 2006.

[T. Hadzic *et al.*, 2006] T. Hadzic, R. Jensen, and H. Reif Andersen. Notes on Calculating Valid Domains. Manuscript online `http://www.itu.dk/~tarik/cvd/cvd.pdf`, 2006.

[Bertoli *et al.*, 2002] P. Bertoli, A. Cimatti, J. Slanley, and S. Thiébaux. Solving power supply restoration problems with planning via symbolic model checking. In *15th European Conference on Artificial Intelligence ECAI'02, Lyon, France*, pages 576–580. IOS Press, 2002.

[Bryant, 1986] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 8:677–691, 1986.

[Dechter, 2003] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, May 2003.

[Diaz *et al.*, 2002] Josep Diaz, Jordi Petit, and Maria Serna. A survey of graph layout problems. In *ACM Computing Surveys*, volume 34, September 2002.

[Dutuit *et al.*, 1997] Y. Dutuit, A. Rauzy, and J.-P. Signoret. Monte-carlo simulation to propagate uncertainties in fault trees encoded by means of binary decision diagrams. In *1st International Conference on Mathematical Methods in Reliability, MMR'97*, pages 305–312, 1997.

[Garey and Johnson, 1979] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[Hadzic and Andersen, 2006] T. Hadzic and H. Reif Andersen. A BDD-based polytime algorithm for cost-bounded interactive configuration. In *Proceedings of The Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, 2006.

[Helmert, 2006] M. Helmert. New complexity results for classical planning benchmarks. In *International Conference on Automated Planning and Scheduling ICAPS'06, The English Lake District, Cumbria, UK, June 2006*, pages 52–61, 2006.

[Jensen *et al.*, 2004] R. M. Jensen, M. M. Veloso, and R. E. Bryant. Fault tolerant planning: Toward probabilistic uncertainty models in symbolic non-deterministic planning. In *Proceedings of the 14th International Conference on Automated Planning and Scheduling ICAPS-04*, 2004.

[Jensen, online] R. M. Jensen. CLab: A C++ library for fast backtrack-free interactive product configuration. `http://itu.dk/people/rmj/clab`, online.

[Meer *et al.*, 2006] E. van der Meer, A. Wąsowski, and H. R. Andersen. Efficient interactive configuration of unbounded modular systems. In *ACM Symposiom on Applied Computing (SAC'06), Dijon, France*, 2006.

[Meinel and Theobald, 1998] C. Meinel and T. Theobald. *Algorithms and Data Structures in VLSI Design*. Springer, 1998.

[Møller *et al.*, 2001] J. Møller, H. R. Andersen, and H. Hulgaard. Product configuration over the Internet. In *Proceedings of the 6th INFORMS*, 2001.

[Subbarayan *et al.*, 2004] S. Subbarayan, R. M. Jensen, T. Hadzic, H. R. Andersen, H. Hulgaard, and J. Møller. Comparing two implementations of a complete and backtrack-free interactive configurator. In *CP'04 CSPIA Workshop*, pages 97–111, 2004.

[Thiébaux and Cordier, 2001] S. Thiébaux and M. O. Cordier. Supply restoration in power distribution systems—a benchmark for planning under uncertainty. In *Pre-Proceedings of the 6th European Conference on Planning (ECP-01)*, pages 85–96, 2001.