Additive Pattern Databases for Decoupled Search

Silvan Sievers[1] Daniel Gnad[2] Álvaro Torralba[3]

[1] University of Basel, Switzerland

[2] Linköping University, Sweden

[3] Aalborg University, Denmark

SoCS, 22nd July 2022

Setting & Motivation

- optimal classical planning as heuristic search
- state of the art: abstraction heuristics
- successful alternative to explicit search: decoupled search

Setting & Motivation

- optimal classical planning as heuristic search
- state of the art: abstraction heuristics
- successful alternative to explicit search: decoupled search
- goal: abstraction heuristics for decoupled search

Background

- planning tasks: finite-domain state variables for representing states

Background

- planning tasks: finite-domain state variables for representing states
- pattern database (PDB) heuristics:
  - project variables to a subset
  - store perfect heuristic values of abstraction

Decoupled Search in a Nutshell

- partition state variables to decompose the task: center factor and leaf factors
- branch over center states and actions, handle leaves separately

Decoupled Search in a Nutshell

- partition state variables to decompose the task: center factor and leaf factors
- branch over center states and actions, handle leaves separately
- decoupled state $s^{\mathcal{F}}$:
  - center state
  - pricing function: cost of reachable leaf states

Decoupled Search in a Nutshell

- partition state variables to decompose the task: center factor and leaf factors
- branch over center states and actions, handle leaves separately
- decoupled state $s^{\mathcal{F}}$:
  - center state
  - pricing function: cost of reachable leaf states
  - $\rightarrow$ represents exponentially many (explicit) member states

Heuristics for Decoupled Search So Far

- given: explicit heuristic h
- given: decoupled state $s^{\mathcal{F}}$
- question: how to use h?

Heuristics for Decoupled Search So Far

- given: explicit heuristic h
- given: decoupled state $s^{\mathcal{F}}$
- question: how to use h?

buy-leaves compilation

- compile prices of $s^{\mathcal{F}}$ into new task
- evaluate h on compiled task

Heuristics for Decoupled Search So Far

- given: explicit heuristic h
- given: decoupled state $s^{\mathcal{F}}$
- question: how to use h?

buy-leaves compilation

- compile prices of $s^{\mathcal{F}}$ into new task
- evaluate h on compiled task

- problems:
  - impractical for abstraction-based heuristics
  - pattern selection based on original task

Alternative Definition for Computing Decoupled Heuristics

contribution: explicit decoupled heuristic

$$h_{\mathcal{F},\text{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + h(s)$$

Alternative Definition for Computing Decoupled Heuristics

contribution: explicit decoupled heuristic

$$h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + h(s)$$

- "best" use of given explicit heuristic
- problem: exponentially many member states

Single PDBs for Decoupled Search

reminder: explicit decoupled heuristic

$$h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + h(s)$$

Single PDBs for Decoupled Search

reminder: explicit decoupled heuristic

$$h_{\mathcal{F},\text{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + h(s)$$

contribution: decoupled PDB

$$\text{dPDB}(h^P, s^{\mathcal{F}}) = \min_{s^P \in S^P} \text{price}(s^{\mathcal{F}}, s^P) + h^P(s^P)$$

Combining Multiple PDBs

explicit search

- given: $\mathcal{H} = \{H_1, \dots, H_n\}$ with H_i additive set of (PDB) heuristics (e.g., disjoint PDBs, cost-partitioned PDBs, etc.)
- canonical combination:

$$h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

Combining Multiple PDBs

explicit search

- given: $\mathcal{H} = \{H_1, \dots, H_n\}$ with H_i additive set of (PDB) heuristics (e.g., disjoint PDBs, cost-partitioned PDBs, etc.)
- canonical combination:

$$h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

- how to transfer to decoupled search?

Naïve Combination

reminder: canonical heuristic

$$h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

Naïve Combination

reminder: canonical heuristic

$$h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

- evaluate PDBs individually (use dPDB to compute $h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}})$):

$$h^{\mathcal{H}}_{\mathcal{F},\text{naïve}}(s^{\mathcal{F}}) = \max_{H \in \mathcal{H}} \sum_{h \in H} h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}})$$

Naïve Combination

reminder: canonical heuristic

$$h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

- evaluate PDBs individually (use dPDB to compute $h_{\mathcal{F},\text{ex}}(s^{\mathcal{F}})$):

$$h^{\mathcal{H}}_{\mathcal{F},\text{naïve}}(s^{\mathcal{F}}) = \max_{H \in \mathcal{H}} \sum_{h \in H} h_{\mathcal{F},\text{ex}}(s^{\mathcal{F}})$$

properties

- information-lossy: use different minimizing member state for each PDB
- inadmissible: may count prices of leaves multiple times in different heuristics

Explicit Decoupled Canonical Heuristic (1)

reminder: explicit decoupled heuristic

$$h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + h(s)$$

Explicit Decoupled Canonical Heuristic (1)

reminder: explicit decoupled heuristic

$$h_{\mathcal{F},\text{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + h(s)$$

- now: $h(s) = h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$

Explicit Decoupled Canonical Heuristic (1)

reminder: explicit decoupled heuristic

$$h_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + h(s)$$

- now: $h(s) = h^{\mathcal{H}}(s) = \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$

contribution: explicit decoupled canonical heuristic

$$h^{\mathcal{H}}_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

Explicit Decoupled Canonical Heuristic (2)

complexity

computing $h^{\mathcal{H}}_{\mathcal{F},\text{ex}}$ is **NP**-complete

Explicit Decoupled Canonical Heuristic (2)

complexity

computing $h^{\mathcal{H}}_{\mathcal{F},\mathrm{ex}}$ is **NP**-complete

- practical implementation via branch-and-bound
- incremental computation of member states allows pruning
- worst case: enumeration of all exponentially many member states

Polynomial-time Approximations (1)

reminder: explicit decoupled canonical heuristic

$$h^{\mathcal{H}}_{\mathcal{F},\text{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

Polynomial-time Approximations (1)

reminder: explicit decoupled canonical heuristic

$$h^{\mathcal{H}}_{\mathcal{F},\mathrm{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

- alternative: consider each $H \in \mathcal{H}$ independently, i.e., move max outward:

$$\max_{H \in \mathcal{H}} \min_{s \in [s^{\mathcal{F}}]} \mathrm{price}(s^{\mathcal{F}}, s) + \sum_{h \in H} h(s)$$

Polynomial-time Approximations (1)

reminder: explicit decoupled canonical heuristic

$$h^{\mathcal{H}}_{\mathcal{F},\text{ex}}(s^{\mathcal{F}}) = \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + \max_{H \in \mathcal{H}} \sum_{h \in H} h(s)$$

- alternative: consider each $H \in \mathcal{H}$ independently, i.e., move max outward:

$$\max_{H \in \mathcal{H}} \min_{s \in [s^{\mathcal{F}}]} \text{price}(s^{\mathcal{F}}, s) + \sum_{h \in H} h(s)$$

- admissible, but lossy approximation

Polynomial-time Approximations (2)

leaf-disjoint (LD) PDBs

additive sets: pairwise leaf-disjoint PDBs

Polynomial-time Approximations (2)

leaf-disjoint (LD) PDBs

additive sets: pairwise leaf-disjoint PDBs

single-leaf (SL) PDBs

each PDB affects at most one leaf

Polynomial-time Approximations (2)

leaf-disjoint (LD) PDBs

additive sets: pairwise leaf-disjoint PDBs

single-leaf (SL) PDBs

each PDB affects at most one leaf

- minimize sum of prices and heuristic separately for each set of affected leaves
- heuristic value equals $h^{\mathcal{H}}_{\mathcal{F},\mathrm{ex}}$

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning
- coverage:

F	
MM	

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning
- coverage:

	explicit search		
		LD	SL
F	284	206	293
MM	**749**	662	743

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning
- coverage:

	explicit search			decoupled search
		LD	SL	expl. dec. heur.
F	284	206	293	212
MM	**749**	662	743	628

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning
- coverage:

	explicit search			decoupled search	
		LD	SL	expl. dec. heur.	LD
F	284	206	293	212	210
MM	**749**	662	743	628	607

Experiments

- PDBs computed with hill climbing and CEGAR
- additivity obtained through saturated cost partitioning
- coverage:

	explicit search			decoupled search		
		LD	SL	expl. dec. heur.	LD	SL
F	284	206	293	212	210	**304**
MM	**749**	662	743	628	607	707

Conclusions

- summary:
  - alternative way of computing explicit heuristics for decoupled search
  - efficient computation of PDBs for decoupled search
  - admissible combination of sets of additive PDBs NP-complete
  - practical implementation and polynomial-time approximations

Conclusions

- summary:
  - alternative way of computing explicit heuristics for decoupled search
  - efficient computation of PDBs for decoupled search
  - admissible combination of sets of additive PDBs NP-complete
  - practical implementation and polynomial-time approximations
- future work:
  - many results independent of type of heuristic: use different abstractions
  - integrate cost partitioning into decoupled search: leaf price partitioning