DecStar

Daniel Gnad[1], **Álvaro Torralba**[2], **Alexander Shleyfman**[3]

[1] Linköping University, Sweden
[2] Aalborg University, Denmark
[3] Bar-Ilan University, Israel
daniel.gnad@liu.se, alto@cs.aau.dk, alexander.shleyfman@biu.ac.il

Abstract

DecStar extends Fast Downward by **Dec**oupled state space search, a technique that exploits the independence between components of a planning task to reduce the size of the state-space representation. Partitioning the state variables into components, such that the interaction between these takes the form of a **Star** topology, decoupled search only searches over action sequences affecting the center component of the topology, and enumerates reachable assignments to each leaf component separately. This can lead to an exponential reduction of the search-space representation. It is not always easy to find a partitioning for a given planning task, though, so we introduce a fallback option which runs explicit-state search whenever no (good) partitioning could be found.

General Overview

Decoupled search reduces the representation size of search spaces by exploiting the structure of the problem within the search (Gnad and Hoffmann 2015; Gnad, Hoffmann, and Domshlak 2015; Gnad and Hoffmann 2018). The size of the *decoupled state space* can be exponentially smaller than that of the explicit state space, which decoupled search achieves by partitioning the task into several components, called *factors*, trying to identify a *star topology* with a single *center factor* that interacts with multiple *leaf factors*. By enforcing this structure, and thereby restricting the dependencies between the components, decoupled search has proven to be very efficient and competitive with state-of-the-art planners.

The performance of decoupled search is highly influenced by the outcome of the *factoring* process, i. e., the process of partitioning the state variables. Just, how to find a good factoring, and what qualifies a factoring as being good? These questions have been addressed by Gnad, Poser, and Hoffmann (2017); Schmitt, Gnad, and Hoffmann (2019); Gnad, Torralba, and Fišer (2022), who devised algorithms that can detect star topologies on a wide range of planning domains. Still, the proposed algorithms can *fail* to find a factoring, or succeed, but return a factoring with undesired properties, e. g., large leaf components that incur a prohibitive runtime overhead when generating new search states. In this case, we simply run explicit-state search instead.

Decoupled search is orthogonal to other known state-space reduction methods, such as partial-order reduction (POR), symmetry breaking, and dominance pruning. Given this orthogonality, decoupled search can and has been combined with these techniques, namely with strong stubborn sets pruning (Gnad, Wehrle, and Hoffmann 2016; Gnad, Hoffmann, and Wehrle 2019), symmetry breaking (Gnad et al. 2017), and dominance pruning (Torralba et al. 2016). POR via strong stubborn sets is a technique that is well-known in explicit-state search and originates from the model checking community (Valmari 1989; Alkhazraji et al. 2012; Wehrle and Helmert 2012, 2014). Symmetry breaking is a widely known approach across many areas of computer science (Starke 1991; Emerson and Sistla 1996; Fox and Long 1999; Rintanen 2003; Pochter, Zohar, and Rosenschein 2011; Domshlak, Katz, and Shleyfman 2012). Dominance pruning identifies states that can be safely discarded, without affecting completeness (and optimality) (Torralba and Hoffmann 2015; Torralba 2017). We use the dominance pruning extensions introduced in Gnad (2021) in addition.

We extend the standard preprocessor of Fast Downward with the h^2-based task simplification by Alcázar and Torralba (2015), which removes irrelevant and unreachable facts and actions from the task.

Implementation & Configurations

Decoupled Search has been implemented as an extension of the Fast Downward (FD) planning system (Helmert 2006). By changing the low-level state representation, many of FD's built-in algorithms and functionality can be used with only minor adaptations. We perform decoupled search like introduced by Gnad and Hoffmann (2018), using the improved *fork* and *inverted-fork* factorings from Gnad, Poser, and Hoffmann (2017), as well as the linear-programming (LP) based bM80s factoring method from Gnad, Torralba, and Fišer (2022). The factoring process is given a time limit of 30 seconds. After FD's translator component finishes, we perform a relevance analysis based on h^2 to eliminate actions and simplify the planning task prior to the search (Alcázar and Torralba 2015).

Decoupled search is the main component of our planner. However, as outlined before, our factoring strategies are not guaranteed to find good task decompositions. Thus, in that case we run explicit-state search as fallback method. Our implementation of decoupled search does not support conditional effects, so we also fall back to explicit-state search in their presence. More advances PDDL features such as de-

Timeout (s)	Search	Heuristics	Notes
300	ADA*	$h^{\text{LM-cut}}$	F, CF, IP, SSS
800	DA*	$h^{\text{LM-cut}}$	LP, SSS, OSS
300	A*	$h^{\text{LM-cut}}$	SSS, OSS
400	A*	blind	-

Figure 1: Portfolio configuration in the optimal track. Components are launched top to bottom.

Timeout (s)	Search	Heuristics	Notes
60	DGBFS	h^{FF}	IF
60	DGBFS	h^{FF}	F, CF, IP
120	DGBFS	h^{FF}	OSS
60	GBFS	h^{FF}	-

Figure 2: Portfolio configuration in the agile track. Components are launched top to bottom.

rived predicates or axioms are not supported by the planner.

Optimal Track In the optimal track, we start by doing the h^2-based relevant analysis for up to 4 minutes. After that, we run a sequential portfolio as specified in Figure 1.

We start with Anytime Decoupled A^* using a fork factoring, the $h^{\text{LM-cut}}$ heuristic (Helmert and Domshlak 2009), cost-frontier pruning (CF) and irrelevance pruning (IP) in the leaf state spaces (Torralba et al. 2016), and do partial-order reduction using strong stubborn sets (SSS) (Gnad, Hoffmann, and Wehrle 2019). We disable the stubborn-sets pruning if less than 30% of the actions have been pruned after 1000 state expansions. For the factoring, we impose a maximum on the domain-size product of variables in a leaf of $L_{max} = 10^6$.

The second component uses decoupled A^* with the LP factoring with $L_{max} = 2^{31} - 1$, the $h^{\text{LM-cut}}$ heuristic, strong stubborn sets pruning, and decoupled orbit-space search (OSS) (Gnad et al. 2017). We disable the stubborn-sets pruning if less than 25% of the actions have been pruned after 1000 state expansions.

The third component is like the second, but performs explicit-state search. The last component is a simple blind search.

Agile Track In the agile track, we do not use the h^2 preprocessor. The portfolio as specified in Figure 2. All configurations run Greedy Best-First Search (GBFS) with the h^{FF} heuristics (Hoffmann and Nebel 2001) and preferred-operator pruning using a dual-queue approach (PO) (Richter and Helmert 2009).

The first configuration uses an inverted-fork factoring with $L_{max} = 10^4$; the second a fork factoring with $L_{max} = 10^4$, cost-frontier pruning and irrelevance pruning; the third config uses the LP-factorings with $L_{max} = 2^{31} - 1$, and orbit-space search. The last component is an explicit-state search.

Satisficing Track In the satisficing track, we run the h^2 preprocessor for 2 minutes. The portfolio as specified in Figure 3. All but the last configurations run GBFS with the h^{FF}

Timeout (s)	Search	Heuristics	Notes
300	DGBFS	h^{FF}	IF
300	DGBFS	h^{FF}	F, CF, IP
800	DGBFS	h^{FF}	OSS
400	GBFS	h^{FF}	-

Figure 3: Portfolio configuration in the satisficing track. Components are launched top to bottom.

heuristics and preferred-operator pruning. The last configuration is a standard LAMA (Richter and Westphal 2010).

The first three configurations run decoupled search in the same way as the agile planner configuration. The only difference is that we use $L_{max} = 10^5$ for the fork and inverted-fork factorings.

Post-Competition Analysis

We analyzed the competition results by investigating the log files that have been provided by the organizers. Table 1 summarizes our findings in a per-domain analysis for all competition domains in all three tracks, optimal (left), agile (middle) and satisficing (right). The first column in each part, $\mathcal{F}$, shows the number of instances in which a factoring was detected, so decoupled search was in play. A first observation is that there are only three domains, quantum-layout, recharging-robots, and ricochet-robots, in which decoupled search can actually be performed. This is a quite low number compared to previous IPCs, so the results are dominated by explicit-state search. The columns SY and S3 show the number of instances in which non-trivial symmetries were detected, respectively in which strong-stubborn sets pruning removed at least one transition in a search space. While strong stubborn sets are not effective in most domains, there are 56 instances in which symmetries can be exploited.

The table shows detailed coverage results, distinguishing the components of each track. We show, from left to right, the components in their order of execution for every portfolio. Hence, for example the "F" column in the optimal track corresponds to decoupled search with ADA* using a fork factoring (cf. Figure 1). We use LM and BL to indicate explicit search with A* and $h^{\text{LM-cut}}$, respectively blind search, and ES to indicate explicit-state search in the agile and satisficing tracks, using GBFS with h^{FF}.

In the optimal track, we see that decoupled search worked well in quantum-layout, solving 12 instances. All but two other instances where contributed by explicit-state search, 17 instances when using $h^{\text{LM-cut}}$, and the majority of 38 instances by blind search. In the agile track, the picture is quite similar, decoupled search does good on quantum-layout, solving 19 out of 20 instances. With LP factoring, it also solves 5 instances in recharging-robots^N. The majority of instances (79) is solved by explicit-state GBFS with h^{FF}. The results in the satisficing track are very close to those in the agile track. Decoupled search performs well in quantum-layout, solves some instances in recharging-robots^N, but explicit-state GBFS with h^{FF} contributes the highest number of instances, with 87 out of 114.

		Optimal Track								Agile Track							Satisficing Track						
Domain	#	$\mathcal{F}$	SY	S3	F	LP	LM	BL	$\sum$	$\mathcal{F}$	SY	IF	F	LP	ES	$\sum$	$\mathcal{F}$	SY	IF	F	LP	ES	$\sum$
folding	20	0	0	0	0	0	2	4	6	0	0	0	0	0	7	7	0	0	0	0	0	6	6
folding^N	20	0	0	0	0	0	2	4	6	0	0	0	0	0	7	7	0	0	0	0	0	6	6
labyrinth	20	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
quantum-layout	20	20	11	5	4	8	0	0	12	20	4	0	13	6	1	20	20	4	0	13	6	0	19
recharging-robots	20	4	0	0	0	1	0	10	11	4	1	0	0	4	7	11	4	1	0	0	4	10	14
recharging-robots^N	20	4	4	2	0	1	4	0	5	5	2	0	0	5	6	11	4	1	0	0	4	9	13
ricochet-robots	20	0	20	0	0	0	4	4	8	17	0	0	0	0	7	7	0	0	0	0	0	10	10
rubiks-cube	20	0	0	0	0	0	0	8	8	0	0	0	0	0	20	20	0	0	0	0	0	20	20
rubiks-cube^N	20	0	0	0	0	0	0	8	8	0	0	0	0	0	20	20	0	0	0	0	0	20	20
slitherlink	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
slitherlink^N	20	0	20	3	0	0	4	0	4	0	0	0	0	0	4	4	0	0	0	0	0	6	6
$\sum$	220	28	56	10	4	10	17	38	69	46	7	0	13	15	79	107	28	6	0	13	14	87	114

Table 1: Per-domain and per-component analysis of the effective methods and coverage in all three tracks. $\mathcal{F}$, SY, and S3 show the number of instances in which a non-trivial factoring was detected, non-trivial symmetries were detected; respectively in which strong-stubborn sets pruning removed at least one transition in a search space. The $\sum$ columns show summed up per-domain coverage for the entire track across components. In all three tracks, the components are listed from left to right in the execution order within the respective portfolio.

Acknowledgments

As we build on Fast Downward, we would like to thank all of its contributors. A big thank you also goes to the competition organizers, Daniel Fišer and Florian Pommerening.

References

Alcázar, V.; and Torralba, Á. 2015. A Reminder about the Importance of Computing and Exploiting Invariants in Planning. In Brafman, R.; Domshlak, C.; Haslum, P.; and Zilberstein, S., eds., *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling (ICAPS 2015)*, 2–6. AAAI Press.

Alkhazraji, Y.; Wehrle, M.; Mattmüller, R.; and Helmert, M. 2012. A Stubborn Set Algorithm for Optimal Planning. In De Raedt, L.; Bessiere, C.; Dubois, D.; Doherty, P.; Frasconi, P.; Heintz, F.; and Lucas, P., eds., *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, 891–892. IOS Press.

Domshlak, C.; Katz, M.; and Shleyfman, A. 2012. Enhanced Symmetry Breaking in Cost-Optimal Planning as Forward Search. In McCluskey, L.; Williams, B.; Silva, J. R.; and Bonet, B., eds., *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*, 343–347. AAAI Press.

Emerson, E. A.; and Sistla, A. P. 1996. Symmetry and Model Checking. *Formal Methods in System Design*, 9(1–2): 105–131.

Fox, M.; and Long, D. 1999. The Detection and Exploitation of Symmetry in Planning Problems. In Dean, T., ed., *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 1999)*, 956–961. Morgan Kaufmann.

Gnad, D. 2021. Revisiting Dominance Pruning in Decoupled Search. In Leyton-Brown, K.; and Mausam, eds., *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*, 11809–11817. AAAI Press.

Gnad, D.; and Hoffmann, J. 2015. Beating LM-Cut with $h^{\max}$ (Sometimes): Fork-Decoupled State Space Search. In Brafman, R.; Domshlak, C.; Haslum, P.; and Zilberstein, S., eds., *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling (ICAPS 2015)*, 88–96. AAAI Press.

Gnad, D.; and Hoffmann, J. 2018. Star-Topology Decoupled State Space Search. *Artificial Intelligence*, 257: 24–60.

Gnad, D.; Hoffmann, J.; and Domshlak, C. 2015. From Fork Decoupling to Star-Topology Decoupling. In Lelis, L.; and Stern, R., eds., *Proceedings of the Eighth Annual Symposium on Combinatorial Search (SoCS 2015)*, 53–61. AAAI Press.

Gnad, D.; Hoffmann, J.; and Wehrle, M. 2019. Strong Stubborn Set Pruning for Star-Topology Decoupled State Space Search. *Journal of Artificial Intelligence Research*, 65: 343–392.

Gnad, D.; Poser, V.; and Hoffmann, J. 2017. Beyond Forks: Finding and Ranking Star Factorings for Decoupled Search. In Sierra, C., ed., *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017)*, 4310–4316. IJCAI.

Gnad, D.; Torralba, Á.; and Fišer, D. 2022. Beyond Stars - Generalized Topologies for Decoupled Search. In Thiébaux, S.; and Yeoh, W., eds., *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling (ICAPS 2022)*, 110–118. AAAI Press.

Gnad, D.; Torralba, Á.; Shleyfman, A.; and Hoffmann, J. 2017. Symmetry Breaking in Star-Topology Decoupled Search. In Barbulescu, L.; Frank, J.; Mausam; and Smith, S. F., eds., *Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS 2017)*, 125–134. AAAI Press.

Gnad, D.; Wehrle, M.; and Hoffmann, J. 2016. Decoupled Strong Stubborn Sets. In Kambhampati, S., ed., *Proceed-*

ings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016), 3110–3116. AAAI Press.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Helmert, M.; and Domshlak, C. 2009. Landmarks, Critical Paths and Abstractions: What's the Difference Anyway? In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 162–169. AAAI Press.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research*, 14: 253–302.

Pochter, N.; Zohar, A.; and Rosenschein, J. S. 2011. Exploiting Problem Symmetries in State-Based Planners. In Burgard, W.; and Roth, D., eds., *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*, 1004–1009. AAAI Press.

Richter, S.; and Helmert, M. 2009. Preferred Operators and Deferred Evaluation in Satisficing Planning. In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 273–280. AAAI Press.

Richter, S.; and Westphal, M. 2010. The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research*, 39: 127–177.

Rintanen, J. 2003. Symmetry Reduction for SAT Representations of Transition Systems. In Giunchiglia, E.; Muscettola, N.; and Nau, D., eds., *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003)*, 32–40. AAAI Press.

Schmitt, F.; Gnad, D.; and Hoffmann, J. 2019. Advanced Factoring Strategies for Decoupled Search Using Linear Programming. In Lipovetzky, N.; Onaindia, E.; and Smith, D. E., eds., *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling (ICAPS 2019)*, 377–381. AAAI Press.

Starke, P. H. 1991. Reachability Analysis of Petri Nets Using Symmetries. *Systems Analysis Modelling Simulation*, 8(4–5): 293–303.

Torralba, Á. 2017. From Qualitative to Quantitative Dominance Pruning for Optimal Planning. In Sierra, C., ed., *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017)*, 4426–4432. IJCAI.

Torralba, Á.; Gnad, D.; Dubbert, P.; and Hoffmann, J. 2016. On State-Dominance Criteria in Fork-Decoupled Search. In Kambhampati, S., ed., *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, 3265–3271. AAAI Press.

Torralba, Á.; and Hoffmann, J. 2015. Simulation-Based Admissible Dominance Pruning. In Yang, Q.; and Wooldridge, M., eds., *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, 1689–1695. AAAI Press.

Valmari, A. 1989. Stubborn sets for reduced state space generation. In Rozenberg, G., ed., *Proceedings of the 10th International Conference on Applications and Theory of Petri Nets (APN 1989)*, volume 483 of *Lecture Notes in Computer Science*, 491–515. Springer-Verlag.

Wehrle, M.; and Helmert, M. 2012. About Partial Order Reduction in Planning and Computer Aided Verification. In McCluskey, L.; Williams, B.; Silva, J. R.; and Bonet, B., eds., *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*, 297–305. AAAI Press.

Wehrle, M.; and Helmert, M. 2014. Efficient Stubborn Sets: Generalized Algorithms and Selection Strategies. In Chien, S.; Fern, A.; Ruml, W.; and Do, M., eds., *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS 2014)*, 323–331. AAAI Press.