Proceedings of the 4th ICAPS Workshop on Hierarchical Planning (HPlan 2021)

Temporal Hierarchical Task Network Planning with Nested Multi-Vehicle Routing Problems — A Challenge to be Resolved

Jane Jean Kiam[1], **Pascal Bercher**[2], **Axel Schulte**[1]
[1]University of the Bundeswehr, Munich, Institute of Flight Systems
[2]The Australian National University, College of Engineering and Computer Science
jane.kiam@unibw.de, pascal.bercher@anu.edu.au, axel.schulte@unibw.de

Abstract

This paper focuses on presenting a complex real-world planning application based on a rescue mission. While temporal hierarchical planning seems to be a promising solution to such a class of problems, given its ability to consider experts' knowledge and dissect the search space, many major challenges of complex real-world planning problems are not addressed yet formally, i.e. recursive decomposition to achieve a goal state, optimization of utility functions defined for abstract tasks, and optimal allocation of tasks to multiple actors.

1 Introduction

Hierarchical planning has proven to be useful in solving many real-world planning problems (Bercher, Alford, and Höller 2019), ranging from web services (Sirin et al. 2004) to medical applications (Fdez-Olivares et al. 2019), robotics (Hayes and Scassellati 2016; Jain and Niekum 2018), and aviation (Benton et al. 2018), to name just a few. Its wide usability owes to its similarity with the task planning humans practise, which usually regards first the planning of tasks at a higher abstraction-level, followed by the refinement of each task down to an executable level. The hierarchical task network planning paradigm can therefore encode naturally known "recipes" for the decomposition of higher-level tasks. Furthermore, by doing so, problem solving is simplified, either by considering a lower-resolution state space at the higher abstraction levels, or by considering a limited search space[1]. This advantage has also been leveraged by Kaelbling and Lozano-Pérez (2011) and Patra et al. (2020), for example to develop hierarchical planners that manage to cope with non-deterministic and dynamic environments. The common driving idea behind is to plan at a more abstract level and leave the details (i.e. lower-level plan) be decided once the knowledge on the environment becomes "clearer". Given their ability to include experts' knowledge and reduce search space, hierarchical planners are promising to solve even more complex real-world planning problems. This paper presents one challenging scenario based on a large-scale rescue mission.

There are many formalisms centered around the idea of hierarchical planning, where "pure" HTN planning is only one, together with other extensions, such as HTN planning with task insertion, Hierarchical Goal Network planning, etc. (Bercher, Alford, and Höller 2019). One of such extensions of major importance for practical applicability of the formalism is featuring time – though only few formalisms and existing planning approaches support it. Works by Goldman (2006) (durative planning with SHOP2), Molineaux, Klenk, and Aha (2010) ($\mathrm{SHOP2}_{\mathrm{PDDL+}}$), Fdez-Olivares et al. (2006; 2019) (complex mission and multi-agent planning), Dvořák et al. (2014) and Bit-Monnot, Smith, and Do (2016) (the FAPE planner), as well as by Stock et al. (2015) (for robotics) are some of these exceptions. Furthermore, a unifying formalism featuring time does not yet seem to be available. In the context of the recent International Planning Competition (IPC) 2020 for HTN planning (Behnke et al. 2019), HDDL (Höller et al. 2020a), the proposed common standard for describing HTN problems, also does not support time (yet). The Action Notation Modelling Language (ANML) (Smith, Frank, and Cushing 2008) focuses on time, but it has not been integrated into HDDL; its semantics is also incompatible with recursion, which is an inherent feature of many HTN planning task models. Besides, its ability to model problems with the need for optimal allocation of tasks to actors, comparable to approaches used for solving multi-vehicle routing problems (MVRPs)[2] is unclear.

For practical real-world use cases we see the support of time as one of the major requirements. This will require both 1) an adequate and easy to use modelling language (potentially complemented with adequate modelling support), and 2) planning systems able to find solutions efficiently, which goes beyond just supporting time technically – it requires new or improved heuristics, or techniques, depending on the respective approach chosen to tackle the problem.

Another challenge that we foresee is how utility functions are being handled, and which kinds of plan metrics are supported. Furthermore, real-world use cases involve often more than one "executor" of the plan, which renders plan-

[1]Search space reduction is possible since HTNs allow to restrict search in directions anticipated by the expert, though in general no bound on the search space exists since the HTN framework allows to express undecidable problems (Erol, Hendler, and Nau 1996).

[2]In a MVRP, vehicles are assigned tasks at different locations (Beck, Prosser, and Selensky 2003). It is often implicitly implied that vehicles with the ability to perform the task are interchangeable, and that each task is assigned to only one vehicle (and will be a goal aimed to reach by the corresponding vehicle).

ning even more complex. Hierarchical planning can be beneficial in this case given its ability to simplify the problem by leveraging experts' knowledge and by dissecting the search space, a concept adopted by Kiam et al. (2019; 2021) to develop a domain-specific planner that computes plans for multiple Unmanned Aerial Vehicles (UAVs) tasked to monitor dissected ground locations in a dynamic environment.

In the following, a real-world temporal hierarchical planning problem (with nested MVRPs) based on a complex rescue mission will be presented. Challenges posed by this class of problem will be analysed and a more formal representation of the problem will be provided to facilitate a more thorough understanding of the problem, allowing thereby the scaling of complexity (i.e. introducing more tasks, actors, complex goal conditions, and utility functions).

2 A Real-World Problem Example

Hierarchical planning is convenient for describing real-world complex planning problems, especially problems that rely on operational rules or experts' knowledge. Rescue missions are a typical example in which leveraging hierarchical planning can be beneficial, as argued in some previous works (Biundo and Schattenberg 2001; Fdez-Olivares et al. 2006; Patra et al. 2020). The section presents a real-world planning problem based on a rescue mission, the challenges of which are either omitted or only partially considered in these works. Fdez-Olivares et al. (2006) consider in their SIADEX planner temporal reasoning. While this is not considered by Patra et al. (2020), the interaction between planning and acting is making it possible to consider the dynamics of the environment.

Figure 1a depicts an example use case of a realistic and complex rescue mission after a disaster (e.g. earthquake) involving multiple rescue teams of different capabilities. Marked in blue is a disaster-struck area[3]; different locations within the area that require rescue operations are marked in orange, while the different objects (e.g. buildings, clusters of victims, etc.) within a location are marked with stars, the sizes of which also indicate the complexity (or rather the duration) of the rescue tasks to be performed. The capabilities of the rescue teams as well as the team members are listed in Table 1. Note that the number of each type of team member (i.e. humans, robots, UAVs) varies in each team.

Part of the task network structure is depicted in Figure 1b. Due to the number of locations (`?l`) to attend to within a disaster area, and the number of objects or patients to cope with at each location, a time-dependent MVRP must be considered in the decomposition of the high-level task `clear-earthquake-disaster(?a)` to decide for the order the locations will be cleared, i.e. the order in which the `clear(?l)` tasks will be performed. Depending on the need, and on the best practices, different tasks are needed for each location (see the two example decompositions of `clear(?l)`, in which one location needs triage `triage(?tt ?l)` and medical aid `aid(?met ?l)` after having monitored the location with `monitor(?mot ?l)`, while the other requires only an infrastructure team to attend to structural damages `build(?it ?l)`). Additionally, as there are several objects within a location, decomposing the task `monitor(?mot ?l)` for example into more "refined" tasks to be performed by the team members (i.e. `patrol(?h ?o)` by a human `?h`, `drive-by(?r ?o)` by a robot `?r`, and `fly-over(?u ?o)` by a UAV) is again a time-dependent MVRP.

[3]There can be more than one disaster areas.

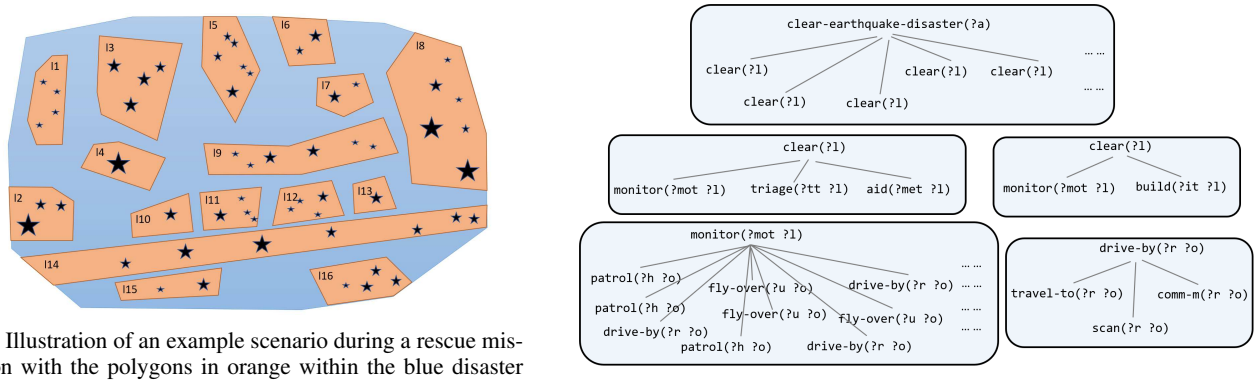
Rescue team	**Team members**
Monitoring, `?mot`	$\{h_1,...,h_H\ u_1, ...u_U, r_1,...,r_R\}$
Triage, `?tt`	$\{h_1,...,h_H, r_1,...,r_R\}$
Medical, `?met`	$\{h_1,...,h_H\}$
Infrastructure, `?it`	$\{h_1,...,h_H, r_1,...,r_R\}$

Table 1: Rescue teams and their heterogeneous capability, as well as the various team members, i.e. robots r_i, human h_i and UAVs u_i

To solve such planning problems, the temporal aspect must be considered, so that the execution of concurrent tasks by different actors are possible, while managing the resources. Besides, such planning problems also pose several new challenges listed below, which, to the best of our knowledge, were either not considered before or do not yet have a straightforward or commonly accepted solution.

- The recursive decomposition into subtasks: The decompositions of the compound tasks, for instance `clear-disaster-area(?a)` and `monitor(?mot ?l)` terminate only if the subtasks altogether achieve the goals imposed by the compound tasks. For `clear-disaster-area(?a)`, the decomposition terminates once all locations `?l` within the area `?a` are cleared, while for `monitor(?mot ?l)`, the decomposition terminates after all objects `?o` within the location are attended to by either the human first responder `patrol(?h ?o)`, a robot `drive-by(?r ?o)` or an UAV `fly-over(?u ?o)`. One way to achieve this is by employing a recursive decomposition, if the number of locations is known at planning time. However, recursive decomposition is not yet very well studied or supported in temporal hierarchical planning. Furthermore, if the exact number of locations is unknown at planning time, such as in a framework where planning and acting interleave[4] (see following subsection), or if the knowledge on the number of locations changes due to updates of information, therefore requiring plan repair (see following subsection), such recursive decomposition is even less straightforward.
- Finding a plan that is optimal with respect to the underlying utility functions: Utility functions such as number of lives saved, cost of structural damages, cost of time, etc. can often only be defined intuitively by the domain expert at a higher abstraction level, i.e. `aid(?met ?l)` for the lives saved and `build(?it ?l)` for the cost of structural damages. However, the exact evaluation of the utility functions can only take place once the decomposi-

[4]This is considered by Patra et al. (2020). However, the temporal aspect is ignored.



(a) Illustration of an example scenario during a rescue mission with the polygons in orange within the blue disaster area depicting the locations where rescue teams are needed.

(b) Decomposition of abstract tasks

Figure 1: An example application for hierarchical planning with nested multi-agent routing problems.

tion is performed down to the level of the primitive tasks. The propagation of utility can therefore be challenging in the formalism for this class of hierarchical planning problems. Furthermore, determining heuristics capable of optimizing different utility functions can also be a challenging issue. The challenge is even more amplified if multiple objectives are to be considered, such as by Kiam, Besada-Portas, and Schulte (2021). Solving it using a hierarchical planning approach is feasible in a domain-dependent manner, but to the best of our knowledge, the feasibility of solving such a class of problems in a domain-independent fashion still remains unknown.

- The allocation of subtasks to actors: This is relevant for example in the decomposition of `monitor(?mot ?l)`, where the subtasks can be allocated to humans `patrol(?h ?o)`, robots `drive-by(?r ?o)` and UAVs `fly-over(?u ?o)`. The optimality of the allocation depends on the utility function; its implementation can be challenging with respect to the formalism or modelling language and the heuristics.

Besides rescue missions, maintenance of a building complex, managing large-scale construction work, or complex logistic problems have similar planning requirements, involving on the one hand hierarchical decomposition of tasks, and on the other, nested time-dependent MVRPs.

Flexible Planning Framework

The above describes only the planning domain and the planning problem in a static fashion. The knowledge of these is insufficient to solve the planning problem, as this must be solved within a framework that adopts a carefully engineered architecture capable of coping with the flow of information on the (dynamic) state space. Concretely, the planning problem can or must be solved in different manners:

- Offline planning: this mode of planning will force the computation of a complete plan with the knowledge assumed before plan execution. Although the plan will be suboptimal since the dynamics of the environment are not considered at planning time, this mode of planning is still useful as an initial overview for resource management.

- Planning and acting: Patra et al. (2020) worked on a planning architecture capable of interchanging between off- and online planning to take into account the dynamics of the environment observed in the course of "acting" (i.e. while carrying out the actions). By doing so, abstract tasks that need not be performed immediately do not have to be decomposed immediately. The decomposition of the task network follows a forward and top-down manner. This kind of framework is convenient when future tasks only need to be planned as an "anticipatory" measure, but detailed action plan is not required immediately.

- Plan and plan repair: This type of planning framework exploits full-fledged offline planning with the information on the state space known before plan execution, yet must be flexible enough to allow for plan repair as new information becomes available during plan execution. However, plan repair is not always straightforward in hierarchical planning, since the execution of a method's task network cannot be aborted and "skip" to the execution of another task network without reversing some effects incurred in the course of the execution of the aborted task network. The permission to "skip" to another task network is often unknown to the planner, as the modelling of all possible plan repair cases would require much more extensive expert knowledge. One possible solution would be to annotate which methods are merely "advice" on how to solve a task (and may be skipped without negative consequences), and which methods actually carry semantics so that a true refinement needs to be found (which implies that it cannot just be skipped). The former repair approach was for example exploited by Goldman, Kuter, and Freedman (2020), while the latter was described by Höller et al. (2020b); but we are not aware of any combination.

3 A More Formal Representation of the Planning Problem

In this subsection, we provide a first attempt for a more formal problem representation of the problem(s) outlined before. However, this does not yet incorporate the dynamics of the world. The planning problem is defined by a tuple $\mathcal{P} = (X_p, X_n, A, A', T_p, T_c, M, s_I, tn_I, G, U, \alpha, \alpha')$, where X_p and X_n are the sets of propositional and numeric state variables respectively, A is the set of actors performing primitive tasks (also called actions), A' is the set of actor-sets, which group actors in teams (e.g. a monitoring team `?mot` is an element of A'), $a' \in A'$ is a set of actors $\{a_1, ..., a_{|a'|}\}$ with $|a'|$ being the cardinality of the set a', $a_i \in A$, and T_p is the set of tuples $(t_p, \delta_{\min}, \delta_{\max})$ containing the primitive task (or action), as well as its minimum and maximum durations respectively, T_c is the set of compound (or abstract) tasks, M is the set of methods to decompose compound tasks into substasks, s_I is the initial state, tn_I is the initial task network, G is the set of goal conditions which define the state(s) to achieve within a time window or before a time instant, and U is the set of utility functions with respect to the (abstract or primitive) tasks. α and α' denote functions that map an actor $a \in A$ (and a set of actors $a' \in A'$ respectively) to a tuple $(t_p, \delta_{\min}, \delta_{\max}) \in T_p$ (and an abstract task $t_c \in T_c$ respectively), where t_p (and t_c respectively) are primitive task (and compound task respectively) that the actor a (and a' respectively) can perform.

A solution to the planning problem is a set of plans, i.e. $\pi = \{\pi^{a_1}, ..., \pi^{a_{|A|}}\}$, where each plan π^a is a partially or totally ordered set of actions with their associated time constraints for actor $a \in A$.

As mentioned in Section 2, depending on the availability of information, the knowledge about the environment can be updated before the planning loop terminates (i.e. in planning and acting), or after the termination of planning and during the plan execution (i.e. plan repair is required). In these cases, this formal representation may no longer be adequate.

4 Expected Features of Future Planners

In this section, expectations on the temporal hierarchical planners to be developed as solutions to the underlying challenges described in Section 2 and 3 will be discussed.

Modelling Language

In order to build on each other's domain-independent reasoning and planning methodologies, which can be based on heuristics, compilation-based or grounding techniques, a common formalism and modelling language are necessary. In view of this, HDDL was developed by Höller et al. (2020a) and used as input language (directly or translated further) for the HTN track of the IPC 2020. However, HDDL does not consider temporal planning aspects. Recent previous works by Fernandez-Olivares and Perez (2020), Bit-Monnot et al. (2020) and Stock et al. (2015) include temporal information; however, they employ different problem modelling languages. Fernandez-Olivares and Perez (2020) use the Hierarchical Planning Description Language (HPDL) for modelling the hierarchical planning problem coupled with a temporal reasoning engine, while Bit-Monnot et al. (2020) use ANML developed by Smith, Frank, and Cushing (2008), and Stock et al. (2015) use a customized domain modelling language that enables the expression of temporal constraints such as start time and duration of tasks.

Therefore, the underlying challenge with respect to the modelling language is to first properly define the formalism for this class of temporal hierarchical planning problems (as described in Section 2) in a more general sense, followed by the development of a syntax capable of taking into account specifically the challenging aspects listed in Section 2, namely the recursive decomposition into subtasks, consideration of utility functions, and the allocation of subtasks to actors (or sets of actors).

Planning System

In AI planning, the development of modelling languages is parallel to the development of the planning system. Besides the qualitative assessment on the adequacy of the modelling language to model the class of planning problems described in Section 2, as well as the correctness of the formalism, a systematic and quantitative assessment of the planning system ought to be considered too. The planning efficiency can be assessed by scaling the problems with respect to the number of tasks, the number of actors, as well as the depth of the task network. Besides, the plan quality can be assessed according to the defined utility functions. These can be linear and non-linear functions, or even more complex mathematical equations such as the Bellman equation considered by Kiam, Besada-Portas, and Schulte (2021), adapted from Boyan and Littman (2000).

Flexible Planning Framework

As described in Section 2, a planning problem can be solved in different manners. Besides the most straightforward offline planning framework, planners must also be compatible with a framework that requires interchanging of off- and online planning to cope with the dynamics of the environment, as well as plan repair, which requires extensive work on the merging of task networks, involving even interaction with human expert(s) to retrieve additional information on the problem models to perform a correct merging.

5 Conclusion

Besides a descriptive overview on a complex temporal hierarchical planning problem, this paper also describes the challenges related to such a class of problems, which must be solved with 1) a proper definition of the formalism, followed by the development of a rigorous problem modelling language, and 2) domain-independent planning systems that support the the modelling language and solve the problem efficiently. For wider usability, the planner must also cope with different planning frameworks, allowing an interleaving of planning and acting, as well as a plan repair.

Our main motivation is to make aware of the unresolved challenges in hierarchical planning that will require collective efforts if such a class of problems is to be solved in a

domain-independent fashion. One way to promote research interest is to gradually include subsets of the aforementioned challenges in future IPC tracks on HTN planning.

References

Beck, J. C.; Prosser, P.; and Selensky, E. 2003. Vehicle Routing and Job Shop Scheduling: What's the Difference? In *Proceedings of the 13th ICAPS*.

Behnke, G.; Höller, D.; Bercher, P.; Biundo, S.; Pellier, D.; Fiorino, H.; and Alford, R. 2019. Hierarchical Planning in the IPC. In *Proceedings of the Workshop on the International Planning Competition*.

Benton, J.; Smith, D.; Kaneshige, J.; Keely, L.; and Stucky, T. 2018. CHAP-E: A Plan Execution Assistant for Pilots. In *Proceedings of the 28th ICAPS*.

Bercher, P.; Alford, R.; and Höller, D. 2019. A Survey on Hierarchical Planning – One Abstract Idea, Many Concrete Realizations. In *Proceedings of the 28th IJCAI*.

Bit-Monnot, A.; Ghallab, M.; Ingrand, F.; and Smith, D. 2020. FAPE: a Constraint-based Planner for Generative and Hierarchical Temporal Planning. *arXiv:2010.13121[cs.AI]* .

Bit-Monnot, A.; Smith, D. E.; and Do, M. 2016. Delete-Free Reachability Analysis for Temporal and Hierarchical Planning. In *Proceedings of the 22nd ECAI*.

Biundo, S.; and Schattenberg, B. 2001. From Abstract Crisis to Concrete Relief – A Preliminary Report on Combining State Abstraction and HTN Planning. In *Proceedings of the 6th European Conference on Planning (ECP)*.

Boyan, J. A.; and Littman, M. L. 2000. Exact Solutions to Time-Dependent MDPs. In *Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS)*.

Dvořák, F.; Bit-Monnot, A.; Ingrand, F.; and Ghallab, M. 2014. A Flexible ANML Actor and Planner in Robotics. In *Proceedings of the 2nd Workshop on Planning and Robotics (PlanRob)*.

Erol, K.; Hendler, J. A.; and Nau, D. S. 1996. Complexity results for HTN planning. *Annals of Mathematics and Artificial Intelligence (AMAI)* 18(1): 69–93.

Fdez-Olivares, J.; Castillo, L.; García-Pérez, O.; and Palao, F. 2006. Bringing Users and Planning Technology Together. Experiences in SIADEX. In *Proceedings of the 16th ICAPS*.

Fdez-Olivares, J.; Onaindia, E.; Castillo, L.; Jordán, J.; and Cózar, J. 2019. Personalized conciliation of clinical guidelines for comorbid patients through multi-agent planning. *Artificial Intelligence in Medicine* 96: 167–186.

Fernandez-Olivares, J.; and Perez, R. 2020. Driver Activity Recognition by Means of Temporal HTN Planning. In *Proceedings of the 30th ICAPS*.

Goldman, R. P. 2006. Durative Planning in HTNs. In *Proceedings of the 16th ICAPS*, 382–385.

Goldman, R. P.; Kuter, U.; and Freedman, R. G. 2020. Stable Plan Repair for State-Space HTN Planning. In *Proceedings of the 3rd ICAPS Workshop on Hierarchical Planning (HPlan)*.

Hayes, B.; and Scassellati, B. 2016. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*.

Höller, D.; Behnke, G.; Bercher, P.; Biundo, S.; Fiorino, H.; Pellier, D.; and Alford, R. 2020a. HDDL: An Extension to PDDL for Expressing Hierarchical Planning Problems. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI)*.

Höller, D.; Bercher, P.; Behnke, G.; and Biundo, S. 2020b. HTN Plan Repair via Model Transformation. In *Proceedings of the 43th German Conference on Artificial Intelligence (KI)*. Springer.

Jain, A.; and Niekum, S. 2018. Efficient Hierarchical Robot Motion Planning Under Uncertainty and Hybrid Dynamics. In *Proceedings of Machine Learning Research - The Conference on Robot Learning (CoRL)*, volume 87.

Kaelbling, L. P.; and Lozano-Pérez, T. 2011. Hierarchical task and motion planning in the now. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*.

Kiam, J. J.; Besada-Portas, E.; Hehtke, V.; and Schulte, A. 2019. GA-Guided Task Planning for Multiple-HAPS in Realistic Time-Varying Operation Environments. In *Proceedings of the 2019 Genetic and Evolutionary Computation Conference (GECCO)*.

Kiam, J. J.; Besada-Portas, E.; and Schulte, A. 2021. Hierarchical Mission Planning with a GA-Optimizer for Unmanned High Altitude Pseudo-Satellites. *Sensors* 21(5).

Molineaux, M.; Klenk, M.; and Aha, D. 2010. Planning in Dynamic Environments: Extending HTNs with Nonlinear Continuous Effects. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*.

Patra, S.; Mason, J.; Kumar, A.; Ghallab, M.; Traverso, P.; and Nau, D. 2020. Integrating Acting, Planning, and Learning in Hierarchical Operational Models. In *Proceedings of the 30th ICAPS*.

Sirin, E.; Parsia, B.; Wu, D.; Hendler, J.; and Nau, D. 2004. HTN planning for Web Service composition using SHOP2. *Journal of Web Semantics* 1(4): 377–396.

Smith, D.; Frank, J.; and Cushing, W. 2008. The ANML Language. In *Proceedings of the Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Stock, S.; Mansouri, M.; Pecora, F.; and Hertzberg, J. 2015. Online task merging with a hierarchical hybrid task planner for mobile service robots. In *Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.