Active Exploitation of Redundancies in Reconfigurable Multirobot Systems

Thomas M. Roehr

***Abstract*—While traditional robotic systems come with a monolithic system design, reconfigurable multirobot systems can share and shift physical resources in an on-demand fashion. Multirobot operations can benefit from this flexibility by actively managing system redundancies depending on current tasks and having more options to respond to failure events. To support this active exploitation of redundancies in robotic systems, this article details an organization model as basis for planning with reconfigurable multirobot systems. The model allows us to exploit redundancies when optimizing a multirobot system's probability of survival with respect to a desired mission. The resulting planning approach trades safety against efficiency in robotic operations and thereby offers a new perspective and tool to design and improve multirobot missions. We use a simulated multirobot planetary exploration mission to evaluate this approach and highlight an exemplary performance landscape. Our implementation of the organization model is open-source available (https://github.com/rock-knowledge-reasoning/knowledge-reasoning-moreorg).**

***Index Terms*—Multirobot systems, planning, reconfigurable robots, scheduling and coordination, space robotics and automation.**

I. Introduction

RECONFIGURABLE multirobot systems introduce a new dimension to the design of future robot missions since they permit robots to exchange physical subsystems. This flexibility to shift subsystems can be exploited to actively manage the level of redundancy of individual robots. This is especially interesting for costly planetary space operations, which require highly redundant robots. The state of the art in planetary space missions is, however, single robotic systems. Despite the fact that international space agencies operate with multiple rovers on the same planet, cooperation between these system has not been targeted. With the consideration of building up infrastructure, creating habitats to prepare human presence and supporting safe operations, this paradigm will have to shift.

Manuscript received May 11, 2021; accepted September 16, 2021. This article was recommended for publication by Associate Editor A. Prorok and Editor P. Robuffo Giordano upon evaluation of the reviewers' comments. This work was supported by the German Space Agency with federal funds of the Federal Ministry of Economic Affairs and Energy for the project TransTerrA under Grant 50RA1301 to implement the initial planning approach. The continued evaluation and application in the project Q-Rock was supported by the Federal Ministry of Education and Research under Grant 01IW18003. The continued development in the project TransFIT was supported by the Federal Ministry for Economic Affairs and Energy under Grant 50RA1701.

The author is with the Robotics Innovation Center, DFKI GmbH, 28359 Bremen, Germany (e-mail: thomas.roehr@dfki.de).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2021.3118284.

Digital Object Identifier 10.1109/TRO.2021.3118284

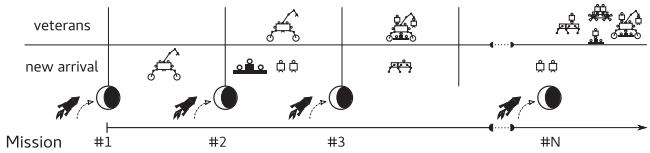


Fig. 1. Schematic description of an incremental design of planetary space missions using reconfigurable multirobot systems.

Current planetary space operations have to rely on ground operators for adaptations and repair, which leads to a very slow and costly process. The dependence on Earth-based maintenance or even hardware deliveries should be minimized for future long-term space missions. An incremental mission design offers an alternative, and the concept is depicted in Fig. 1. Here, not only software, but also hardware subsystems can evolve with the experience made in previous missions and incrementally improve already operating multiagent systems.

The possibility to extend or refurbish existing hardware is a significant advantage, even more when a team of robots can perform this process autonomously. The autonomous exploitation of features of a reconfigurable multirobot system is, nonetheless, a significant challenge, including practical issues regarding distributed communication and planning approaches. Additionally, an application in a space context requires risk mitigation strategies and high safety standards. Therefore, enabling planning approaches that permit to exploit redundancy and sharing of resources between robotic systems will be one step forward toward safe long-term operations of autonomous multirobot systems. By introducing a Model for Reconfigurable Multirobot Organizations (MoreOrg) in this article, we offer a modeling approach with focus on (physically) reconfigurable multirobot systems, although the model can embed classical nonreconfigurable robots. This enables a planning approach that exploits reconfigurability, as described in Section IV. The focus of this article is, however, on the design and role of the organization model in this context.

The organization modeling and planning approach for reconfigurable multirobot systems finds its initial motivation in space applications. In this article, we also refer to this context to provide an exemplary use of our suggested modeling approach. Meanwhile, the organization model builds on ontologies and is, therefore, extensible, so that users can add new subsystems, functionalities, robots, custom properties, and inference rules by extending the ontological description.

A. History and Related Work

Implementations of reconfigurable multirobot systems exist within a spectrum ranging from industrial robots, which allow an end-effector exchange to fully self-reconfigurable multirobots systems [1]–[3]. Research in the area of reconfigurable multirobot systems has initially been driven by the latter, i.e., the concept of the so-called self-reconfigurable systems. Their main design characteristic is a high level of redundancy of mostly homogeneous modules, which can automatically restructure to establish a target structure; a broad review of self-reconfigurable multirobot systems is provided by Chennareddy *et al.* [4] and Liu *et al.* [5]. Self-reconfiguration aims a providing highly resilient systems, i.e., being able to recover from disruptive (structural) changes and failures. These highly modular systems typically suffer from limited capabilities and thus lack broad applicability. Planning approaches in this context focus on the transition between two organization states, e.g., Baca *et al.* [6] apply coalition game theory to optimize the organizational state. They characterize coalitions, however, based on strong assumptions with a utility function, which 1) has a static utility for each agent independent of the coalition it will be embedded into and 2) cannot account for interface compatibility issues leading to constrained coalition formation. The swarm-bot system developed by Mondada *et al.* [7] initially takes a middle ground and uses a simple structured yet reconfigurable swarm-based system in combination with a behavior-based control approach to exploit reconfigurability. They are able to illustrate team capabilities that arise from superaddition such as gap and obstacle traversal as well as (heavy) object transport.

Similarly to the swarm-bot system, our work targets reconfigurable multirobot systems, which consist of individual agents that can already be considered capable robots. Wilcox *et al.* [8], for instance, developed the reconfigurable six-legged robot ATHLETE to support infrastructure buildup on planetary surfaces. Although Wilcox *et al.* [8] did approach automation of reconfiguration procedures, they did not develop high-level planning approaches to fully exploit reconfigurability. For similar capable robots, reconfigurability focuses on the adaption of internal subsystems, e.g., the Scarab rover [9] is able to adapt its locomotion platform. Reid *et al.* [10] show how to exploit this kind of reconfigurability with a dedicated sampling-based (motion) planning approach after modeling the reconfiguration space of the motion planning system.

In the context of organization research outside of the robotics domain, Dignum [11] looks at reconfiguration of organizational processes involving humans. According to Dignum, the general need to actively organize teams aims at increasing efficiency, and she sees flexible and adaptive organizations as suitable means to deal with dynamic environments. She suggests that organizations conditionally adapt and should reorganize if this will lead to an increasing success of an organization; even a suboptimal reorganization can be better than no response at all. Still, the question when to reorganize and when to accept loss is left unanswered.

In her work, Dignum points to *strategic flexibility*, a concept developed in the scope of managing high-technology industries by Evans [12]. Evans' framework conceptualizes the strategic use of a company's or more generically a market player's flexibility. Flexibility to adapt leads to a significant competitive advantage, since it offers a market player additional means to encounter unforeseen events. Hence, adaptation can directly lead to a greater probability of survival or net monetary benefit for market players. In his work, Evans [12] refers to proactive, reactive, defensive, and exploitative system capabilities and relates defensive ones to robustness and resilience. While robustness refers to a system that can endure impacts up to a certain degree without breaking, resilience results from the ability to recover from error and return into a functional state.

Especially, resilience is a key to survival, not only in natural systems, but also for technical and social systems alike shown by examples collected from [13]. Resilient systems rely on their capability to adapt. Therefore, reconfigurability can contribute to an increased resilience of robotic systems. Evans' conceptual framework is general enough to be applied to reconfigurable multirobot systems, and his categorization of maneuvers can be similarly applied for a characterization of robotic activities: protective and corrective activities count as defensive maneuvers.

The design of a space robot is typically focusing on defensive measures by adding redundancies and preparing failure handling strategies. What the flexibility of reconfigurable multirobot systems offers, after all, is the possibility for an active management of these redundancies, for instance, to adapt the organization to respond to functional requirements or to optimize the redundancy level across all available systems. An active management with a global optimization policy will treat all resources equally. This means that the controller of a robotic mission can influence the redundancy level of resources. As a side effect, active management might even result in an overall cost-optimized system design, by reducing the mean redundancy level of the multirobot system.

Continuous optimization of an organization structure can also be found in Model of Organization for multIagent SystEms (MOISE+). Hübner *et al.* [14] focus on a design pattern to control the reconfiguration process and identify key components. They outline an architecture to continuously optimize an organization structure to achieve main organization's objectives. Objectives for a (sub)team can be defined as a hierarchical task network in a so-called scheme, which leads to the definition of a behavioral pattern, e.g., they use playing soccer as primary example. A reconfiguration process or rather transition from one team structure to another can be planned or unplanned: planned transitions can be triggered in a top-down fashion by an external operator or they can be scheduled for a specific time. Hübner *et al.* [15] require planned transitions to follow a previously defined and, therefore, static reorganization pattern, while unplanned transitions have to be dynamically controlled by agents.

The reorganization process in MOISE+ itself requires forming a special predefined group structure: one agent has to adopt the role of the so-called *OrgManager* in order to organize the overall reconfiguration. The reconfiguration group also requires at least one agent to take over the *Designer* role, in order to analyze the current status of the organizational structure, and suggest a potentially better structure.

In the area of robotics, Organization Model for Adaptive Computational Systems (OMACS) is another approach for designing an organization model presented by DeLoach *et al.* [16], [17]. The main concepts in OMACS are goals, roles, agents, and capabilities. OMACS uses a capability-based representation for a role, i.e., a role is defined by a set of capabilities. The quality of an agent's capability can be quantified using normalized values. In the same way, DeLoach [17] quantifies an agent's ability to fulfill a role based on its capabilities. OMACS sets the main focus on the quantification of the potential of abstract roles and agents to contribute toward an organization's success. The usage of this information allows us to optimize the team structure by allowing the best suited agent to handle a task and thereby increases the likelihood of an organization's success. Similar to MOISE+, DeLoach [17] suggests the use of behavior policies to control the cooperative behavior of agents. In OMACS, an organization designer can explicitly define reorganization rules. For instance, to specify if and how one agent can replace another agent once the latter becomes unable to fulfill a role. An application of runtime reorganization has been shown with three real robots and a single laptop agent by Zhong and DeLoach [18]. OMACS assumes atomic capabilities without composition, and the value normalization to [0, 1] restricts the quantification, e.g., for a qualification of capability, to a single dimension. The quality of an agent's capability has, therefore, (initially) unclear semantics, which limits the applicability of the approach in practical applications.

Our approach looks superficially very similar to MOISE+ and OMACS with respect to exploiting a mapping between structure and function. However, MOISE+ and OMACS do this still on a higher level: MOISE+ simply defines the suitability of an agent to fill a role, while OMACS already analyzes agent capabilities. Both organization models fit loosely coupled agent teams, but they neither take physical reconfiguration under resource constraints into account nor can infer functionality of newly composed agents. While MoreOrg takes a similar capability-based approach of identifying an agent's available functionality to OMACS, it: 1) derives this information dynamically from the available set of hardware and software resources, and thus permits inference of properties; and 2) does not (yet) characterize the quality of a function. Instead, we estimate the probability of survival of a function, based on the available resources. Neither MOISE+ nor OMACS offers multiagent planning; instead, they allow one to control agent behavior via predefined tasks.

The robotic framework KnowRob [19] uses a semantic modeling approach and uses ontologies to represent the structure of a robotic system to create a mapping between structure and function. Beetz *et al.* [20] exploit this abstraction by defining plan templates for a single robot, which can then be used to identify required functionality during plan execution.

In contrast to the existing robot modeling and planning approaches, MoreOrg can model a heterogeneous set of physically reconfigurable robots and infer agent as well as organization properties. This permits an exploitation of superadditive effects and in parallel accounts for safety. To the best of our knowledge, none of the existing organization modeling and planning approaches in robotics has covered these aspects so far.

B. Relation to Previous Work and Contribution

We base the results in this article on the practical experience gained from working with multiple teams of reconfigurable systems [21]–[24].[1] Furthermore, this work is part of the planning approach developed with a special focus on reconfigurable multirobot systems [25]–[27]. The contributions of this article are the following: 1) detailing an organization model for reconfigurable multirobot systems with focus on functionality-based probability of survival; 2) offering the open-source implementation of this model; and 3) evaluating the tradeoff between safety, efficacy, and efficiency for an exemplary space mission.

C. Outline

This article takes a bottom-up approach in describing MoreOrg. In Section II, we first provide our revised formalization and terminology for reconfigurable multirobot systems, including atomic and composite agents. Section III extends the modeling and formalization to agent and organization properties and in particular how they can be generically inferred. Redundancy is a special property and quantified in this context with respect to required functionality. In Section IV, we detail our planning or rather optimization approach for reconfigurable multirobot systems that is based on MoreOrg. In Section V, we describe an exemplary planning result. In Section VI, we discuss the current state and open challenges and give a critical review on our take on dealing with reconfigurable multirobot systems.

II. Reconfigurable Multirobot Systems

This section provides the basic notation, definitions, and the underlying assumptions regarding reconfigurable multirobot systems. The notation builds on the formalisms found in coalition games [28]. In particular, the agent-type representation is based on the representations developed by Shrot *et al.* [29] and Ueda *et al.* [30].

While reconfiguration affects hardware and software alike, the focus of this work is on handling physical reconfiguration of agents. The level of granularity is chosen correspondingly with the lowest level being a physical agent, which cannot be separated further into two or more physical agents. This agent is denoted by *atomic agent*.

Definition (Atomic agent): An *atomic agent* a represents a monolithic physical robotic system.

Note that a physical agent representing an atomic agent still contains subsystems. They are, however, inseparable parts of the physical agent.

Definition ((Atomic) Agent pool): An *agent pool* A denotes a set of atomic agents, such that $A = \{a_1, \ldots, a_{|A|}\}$ is the set of all atomic agents, and $a \in A$ or equivalently $\{a\} \subseteq A$. A set of agent pools is denoted by $\mathbf{A} = \{A_1, \ldots, A_{|\mathbf{A}|}\}$.

Connection interfaces are the key elements in a reconfigurable system and, here, open the opportunity for combining two or more atomic agents. A composition from two or more atomic agents is referred to as *composite agent*. The join operator $\cup$ in

[1] Field Trials Utah: [Online]. Available: https://www.youtube.com/watch?v=pvKIzldni68

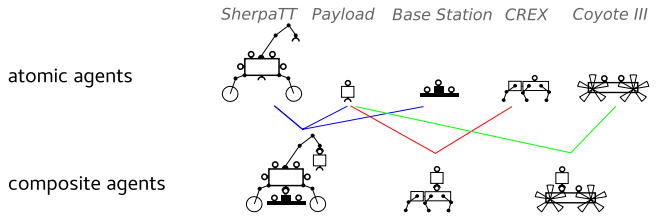


Fig. 2. Available set of atomic agents and a subset of composite agents that can be formed by combining different atomic agents (see [23] and [24] for details on the real counterparts). From top left to right: 1) a rover with four male and two female interfaces; 2) a payload item with one male and one female interface; 3) a base station with four male interfaces; 4) a legged robot with one male interface; and 5) a star-wheeled robot with two male interfaces.

the following definition for composite agents aligns well with the actual physical join operation of atomic agents and permits an intuitive representation.

Definition (Composite agent): A linked system of two or more atomic agents is a set CA, which is denoted by *composite agent* $CA = a_i \cup \cdots \cup a_j = \{a_i, \ldots, a_j\}$, where $a_i, \ldots, a_j \in A, |A| \geq |CA| > 1$.

Fig. 2 illustrates the general approach to agent composition as basis for superaddition, as explained with the following example: A mobile robot (atomic agent m) can share its power source with other robots, but it has no camera. After attaching an unpowered atomic agent c, which has one camera as a subsystem, the composite agent $\{m, c\}$ is equipped to take images. It can now move to any location and take images—a functionality neither of the atomic agents m or c provides.

Combinatorial explosion is one of the main challenges to deal with when considering a reconfigurable system with a large number of atomic agents. One means to reduce the effects of combinatorial explosion is typing. Agent typing allows dealing with same typed agents using homogeneously formed partitions of an overall set of agents.

Definition (Atomic and composite agent type): The type of an atomic agent a is denoted by $\hat{a}$, and equivalently for a composite agent CA, the type is denoted by $\widehat{CA}$. The set of all atomic agent types is denoted by $\theta(A) = \{1, \ldots, |\theta(A)|\}$, with the corresponding type-partitioned sets of agent instances $A^1, \ldots, A^{|\theta(A)|}$, where $A = A^1 \cup \ldots \cup A^{|\theta(A)|}$ and A^x represents an agent pool containing only atomic agents of type x.

The concept of a (general) agent wraps the concepts of atomic and composite agents. Henceforth, the term agent is equivalently used to the term general agent.

Definition (General agent): Any subset $A' \subseteq A$, where $A' \neq \emptyset$ forms a physical coalition, is denoted by *general agent*. A (general) agent has a corresponding atomic agent-type partitioned set of agent instances $A^1, \ldots, A^{|\theta(A)|}$, where $A = A^1 \cup \ldots \cup A^{|\theta(A)|}$.

Definition (General agent type): The type of a (general) agent GA is denoted by $\widehat{GA}$. A general agent type $\widehat{GA}$ is represented as a function $\gamma_{\widehat{GA}} : \theta(A) \rightarrow \mathbb{N}_0$. The function $\gamma_{\widehat{GA}}$ maps an atomic agent type $\hat{a}$ to the cardinality $c_{\hat{a}}$ of the type partition of $\widehat{GA}$, such that $c_{\hat{a}} = |GA^{\hat{a}}|$. Equivalently to $\gamma_{\widehat{GA}}(\hat{a}) \geq 1$, the following notation will be used: $\hat{a} \in \widehat{GA}$, and $\hat{a} \notin \widehat{GA}$ for $\gamma_{\widehat{GA}}(\hat{a}) = 0$.

The reverse mapping from type $\widehat{GA}$ to the general agent is denoted by $i(\widehat{GA}) = GA$.

A general agent type is also represented as a collection of tuples relating agent type and cardinality: $\{(\hat{a}_0, c_{\hat{a}_0}), (\hat{a}_1, c_{\hat{a}_1}), \ldots, (\hat{a}_n, c_{\hat{a}_n})\}$.

An agent pool can now be represented in two ways: as set of atomic agents as already introduced or as general agent type $\widehat{A}$, such that $\forall a \in A : \gamma_{\widehat{A}}(\hat{a}) = |A^{\hat{a}}|$, where the latter offers a more compact representation and is used preferably in our implementation of the organization model.

To execute robotic missions, atomic agents from an available agent pool will be assigned to particular tasks. However, if multiple atomic agents of the same type exist and equal start conditions hold for these atomic agents, multiple equivalent assignments of atomic agents to a task are possible. For that purpose, requirements for atomic agents will be defined by the so-called roles, which act as correctly typed placeholders for instances of an agent type.

Definition (Atomic agent role): An *atomic agent role* $r^{\hat{a}}$ represents an anonymous agent instance of an atomic agent type $\hat{a}$.

Given an overall set of atomic agents, various reconfiguration states of the overall systems are possible. These reconfiguration states result from forming different sets of composite agents, but always with the restriction of the overall available set of atomic agents. In the field of multiagent systems and particularly characteristic function games (see [28]), this leads to the so-called coalition structures. A coalition structure represents the set of active atomic and composite agents that form a reconfigurable multirobot system. Note that we will also use the term *organization* in order to describe a reconfigurable multirobot system represented by an agent pool A.

Definition (Coalition structure): A coalition structure of an agent pool A is denoted by CS^A and is represented by a set of disjunct general agents $CS^A = \{GA_0, \ldots, GA_n\}$, where $GA_0 \cup \ldots \cup GA_n = A$, and $i, j = 0, \ldots, n, \forall i, j, i \neq j : GA_i \cap GA_j = \emptyset$.

Composite agents result from the combination of atomic agents. We use the following definitions to separate the current (realized and physically assembled) set of general agents in a coalition structure from the (virtual) set of agents, which can be formed from the set of atomic agents.

Definition (Operative and dormant agents): Let the current state of a reconfigurable multirobot system be described by a coalition structure CS^A. Then, all general agents $GA \in CS^A$ are referred to as *operative agents*, and complementary, all general agents $GA \in \mathcal{P}^A \wedge GA \notin CS$ are referred to as *dormant agents*, where $\mathcal{P}^A$ is the powerset of all atomic agents.

The previous definitions look at a reconfigurable multirobot system as a collection of agents and consider pairing and coalitions only at this level of modularity. A reconfigurable multirobot system can form composite agents in different ways depending upon the compatibility of interfaces. Hence, to perform a detailed reasoning on connectivity of agents, we also have to account for the physical interfaces as subsystems of an agent to analyze the feasibility of all agents. The scope of the presented formal

description so far is based on a set-theory description and covers what is denoted agent space.

Definition (Agent space): Agent space denotes the set-theory-based view to a reconfigurable multirobot system without constraining the connectivity between any two agents.

Agent space is only a restricted view onto link space.

Definition (Link space): Link space denotes a graph-based structure of a reconfigurable multirobot system. In link space, a reconfigurable multirobot system is represented by an undirected graph $G = (V, E)$, where each vertex $v \in V$ maps to an atomic agent's interface and an edge $e = (u, v)$, $u, v \in V$ represents the existing connection between two interfaces.

The current modeling approach regarding link space accounts only for edges that represent electromechanical connections between agents as precondition for sharing resource in a composite agent. Data connections between software and hardware components as well as configuration options of hardware and software components are currently left out in our modeling approach.

A. Assumptions

A large spectrum of reconfigurable multirobot systems exists. Most often, fully distributed control approaches apply, due to the use of swarm-based systems. The definition of the general agent already reflects one important design choice of this work, which relaxes this apparent requirement for distribution. Instead of enforcing distributed control approaches at all system levels, centralized control approaches for locally autonomous and self-sustained operation of agents are permitted and feasible. This implicitly allows an atomic agent to act as a temporary "master" in a master–slave architecture. When forming a composite agent, for instance, a single atomic agent in this formation acts as master, which is able to control all other attached atomic agents. In effect, each general agent represents a single-minded (collaborative) agent. The distribution of the overall agent system is still maintained by an appropriate design of the operational infrastructure.

Assumption (Individual agent): Each atomic and composite agent comprises a central controller and thus represents an individual single-minded agent.

Generally, two atomic agents can connect via multiple interfaces. We assume, however, limited connectivity and currently do not consider geometrical constraints. This restriction allows us to focus on the identification of essential needs for modeling and automating of reconfigurable multirobot systems.

Assumption (Single link): A physical coupling between two atomic agents can only be established through two and only two compatible coupling interfaces.

In principle, Definition II allows a single agent to have multiple types. However, we assume a single characterizing agent type, which represents the combination of all its parent types or a specialization thereof. This means that one agent type can still inherit the properties of multiple parent types.

Assumption (Single agent type): An agent can be mapped to a single agent type only.

Assumption (Agent type inheritance): An agent type can inherit the properties of another agent type.

A key assumption, when dealing with an active exploitation of resource, is the possibility to join the available resources of two or more agents. Hence, when two or more atomic agents form a composite agent, they join their set of resources. In principle, geometrical restrictions might apply to reuse the set of resources effectively. However, we initially assume that resources are shared without restriction within a composite agent.

Assumption (Resource usage): A composite agent can reuse the subsystems of its composing atomic agents.

To enable resource sharing in a composite agent, various ways of coupling two or more atomic agents can be considered, e.g., electromechanical or thermoelectromechanical. We currently assume, however, that composite agents establish links between their composing atomic agents, which permit data, energy, and power transmission.

Assumption (Agent linkage): Links that connect atomic agents in a composite agent permit transfer of data, energy, and power.

To effectively exploit resources in a redundant structure, the following assumption is made.

Assumption (Component substitution): To maintain the functionality of an agent, one component can replace another if it is an instance of the other's class, which also includes instances of subclasses.

This seems like a strong assumption, since even if components are instances of the same concept, e.g., a camera, it might not be possible to substitute one with the other without losing functionality. However, this is a matter of modeling equivalence as part of the ontological design in MoreOrg.

III. Organization Modeling

We have developed the organization model MoreOrg[2] to quantify the properties of a reconfigurable multirobot system and provide cost measures for a reconfigurable multirobot system. The organization model permits a quantification of system properties at different granularity levels and can characterize the active set of agents, i.e., the coalition structure of the organization by using a bottom-up approach. Fig. 3 depicts the hierarchical decomposition of an organization, which serves as baseline for MoreOrg's reasoning approach. The coalition structure of a reconfigurable multirobot system can change on-demand, which might involve creating and/or removing one or more connection between atomic agents, but all agents can be characterized by their set of associated resources, i.e., hardware and software components.

MoreOrg relies on ontologies to describe resources in general and more specifically atomic agents and their associated functionalities and subsystems. All subsystems and atomic agents are characterized by data properties, e.g., MoreOrg focuses on a set of numeric data properties to enable mobile transport agents (cf. [26], where we relate planning for reconfigurable multirobot systems to vehicle routing problems). As a benefit of this ontological

[2][Online]. Available: https://github.com/rock-knowledge-reasoning/knowledge-reasoning-moreorg

TABLE I
STATIC ATOMIC AGENT TYPE PROPERTIES (ADAPTED FROM [26])

Property	Syntax	Description
velocity	$v_{nom}(\widehat{a})$	nominal velocity of an agent type $\widehat{a}$, $\lvert v_{nom}\rvert > 0$ for mobile atomic agent types and $v_{nom} = 0$ for immobile
transport capacity	$tcap(\hat{a})$	maximum total capacity of an agent of type $\hat{a}$ to transport other agents
transport consumption	$tcon(\hat{a})$	number of storage units an agent of type $\hat{a}$ consumes, when being transported by another agent ($tcon$ is set to 1 for all agent types if not mentioned otherwise)
transport load	$tload(a)$	current load transported by an atomic agent a, i.e., represents the consumed transport capacity of an agent
power source capacity	$esourcecap(\hat{a})$	power source capacity of an atomic agent in Ah
supply voltage	$esupply(\hat{a})$	electrical supply voltage of an atomic agent in V
power consumption	$pw(\hat{a})$	(electrical) power consumption of an agent of type $\hat{a}$

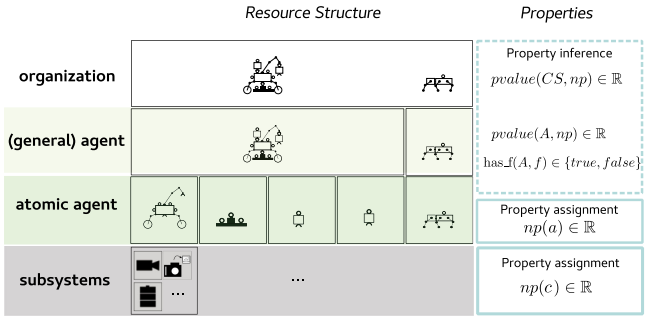


Fig. 3. Organization model is based on a hierarchical view and corresponding property generation. Atomic agents come with (mainly) static properties assignments, while composite agents and overall organization properties have to be dynamically derived from the active coalition structure.

representation, description logic (DL)-based reasoning can be applied and an agent's available functionalities and properties can be inferred from its composing set of resources or rather subsystems. Additional reasoning mechanisms are applied to finally describe the organizational properties, where our focus is set on efficacy and safety. The following subsections detail the organization model and its reasoning approach.

A. Subsystem Properties

Subsystems are tightly bound to atomic agents and come with statically defined properties. All subsystems are at least associated with a *probability of survival* to define their reliability. This is the basis for computing a safety measure for the overall multirobot organization. The details of this computation are provided in Section III-D.

B. Generic Agent Type Properties

MoreOrg provides a mechanism to define agent properties in a general way, such that composite agent properties can be derived from the atomic agents that form the agent.

Definition (Agent type property value): The value function for an agent type $\widehat{A}$ and a numeric property named np is denoted by $pvalue(\widehat{A}, np)$.

Some atomic agent type properties as listed in Table I are statically defined and required to implement the planning approach. Note that other uses of the organization model might require a completely different set of properties. Hence, the listed properties are examples only and have been added to support the operation of a logistics chain as targeted reconfigurable multirobot planning problem in space exploration (see Section IV). Nevertheless, the already defined properties will likely be sufficient and needed for many standard robotic scenarios. Any missing properties can easily be added to the ontology if needed.

While many properties of atomic agent types are directly set, some atomic agents' properties and all properties of composite agents have to be inferred. Here, MoreOrg distinguishes between Boolean and numeric properties (cf. Table II).

1) Boolean Properties: Boolean properties map to the availability of particular capabilities, and they can be inferred from available combinations of functionalities and subsystems, e.g., the functionality *AutonomousNavigation* is inferred from the availability of other capabilities and subsystems, here *Locomotion*, *Mapping*, *MotionPlanning*, *SelfLocalization*, and a subsystem *PowerSource*. The Boolean property $has\_f(\widehat{A}, f)$ defines whether an agent A supports a functionality f or not. This Boolean property enables selection mechanisms, e.g., to identify agents which have the functionalities to perform a particular tasks.

For atomic agents, the inference of available functionality is based on ontological reasoning and exploits available DL-based reasoners, here Fact++ [31]. For composite agents, we allow us to quantify the *support* for a functionality by an agent's resources, so that the Boolean property depends on the amount of support:

$$has\_f(\widehat{A}, f) = \begin{cases} \text{true}, & \text{if } support(\hat{A}, f) \geq 1 \\ \text{false}, & \text{otherwise} \end{cases} \quad (1)$$

where $\widehat{A}$ is the agent type, f is a functionality, and $support$ is defined in the following.

Support for an agent's functionality is based on a single resource concept c, e.g., where c can represent a subsystem type such as *Camera*, as follows (see also [27]):

$$support(\widehat{A}, c, f) = \begin{cases} 0, & \text{if } card_{\min}(c, f) = 0 \\ \frac{card_{\max}(c, \widehat{A})}{card_{\min}(c, f)}, & \text{otherwise} \end{cases} \quad (2)$$

TABLE II
INFERRED AGENT TYPE PROPERTIES

Property	Syntax	Description
has functionality	$has\_f(\widehat{A}, f)$	boolean property, defines whether an agent of type $\hat{A}$ has functionality f. The truth value is inferred from the resource dependencies that are defined for f in the ontology.
efficacy	$\mathit{efficacy}(\widehat{A}, \mathcal{F})$	boolean property, defines whether an agent type $\widehat{A}$ supports all functionalities in $\mathcal{F}$ or not
numeric property	$pvalue(\widehat{A}, np)$	numeric property, value for an agent of type $\widehat{A}$ and a property np
reliability	$R(\widehat{A}, \mathcal{F})$	numeric property, reliability ([0,1]) of an agent type $\widehat{A}$ with respect to functionalities in $\mathcal{F}$, based on resource redundancies
operation cost	$ocost(\widehat{A}, t)$	numeric property, here: total consumed power for an agent type $\widehat{A}$ over time t

where $card_{\min}$ and $card_{\max}$ return the minimum required and maximum available cardinality of resource instances (including instances of derived resource concepts), respectively. Accordingly, support of a functionality f with respect to a resource class c can be categorized as follows:

$$support(\widehat{A}, c, f) = \begin{cases} 0, & \text{no support} \\ \geq 1, & \text{full support} \\ > 0 \text{ and } < 1, & \text{partial support} \end{cases}. \quad (3)$$

Support for a single functionality and subsequently for a set of functionalities $\mathcal{F}$ is then defined as

$$support(\widehat{A}, f) = \min_{c \in \mathcal{C}} support(\widehat{A}, c, f) \quad (4)$$

where $\mathcal{C}$ is a set of resource concepts and $\forall c \in \mathcal{C} : card_{\min}(c, f) \geq 1$ to account only for relevant resource concepts, and

$$support(\widehat{A}, \mathcal{F}) = \min_{f \in \mathcal{F}} support(\widehat{A}, f). \quad (5)$$

2) Numeric Properties: Not all static numeric properties of atomic agents need to be directly assigned, since they can be inferred from other properties, e.g., due to laws of physics. As an example, available energy capacity (Wh) can be derived from the capacity of the power source (Ah) and the supply voltage (V):

$$pvalue(A, ecap) = pvalue(A, esourcecap) \cdot pvalue(A, esupply)$$

where $|A| = 1$. MoreOrg allows us to define these mathematical relationships between properties. This is done by annotating properties in the ontology and by using a simple domain-specific language in combination with a math parser library.[3]

To infer the value of numeric properties for composite agents, MoreOrg permits the definition of custom inference rules. These rules are defined as higher order functions, which can be constructed from selection policies and composition operations.

a) Selection policy: A selection policy $A' = sel(A)$ permits to identify a subselection of atomic agents A' from a (general) agent A according to defined criteria. It is built from subselection policies, which take the form: $\mathbf{A}' = sel(\mathbf{A}, \dots)$.

[3]muParser: [Online]. Available: https://beltoforion.de/en/muparser

MoreOrg offers three basic subselection policies to build custom selection policies:

1) agent size-based selection: $\mathbf{A}' = size\_sel(\mathbf{A}, op, \beta)$, where $\forall A \in \mathbf{A}' : |A|\ op\ \beta$ with $op \in \{<, <=, >, >=, =\}$;
2) functionality-based selection: $\mathbf{A}' = func\_sel(\mathbf{A}, f)$, where $\forall A \in \mathbf{A}' : has\_f(\widehat{A}, f)$;
3) property-based selection: $\mathbf{A}' = prop\_sel(\mathbf{A}, op, np)$, where $\forall A \in \mathbf{A}' : A = op_{A \in \mathbf{A}}\ \mathit{pvalue}(\widehat{A}, np)$, where $op \in \{\text{argmax}, \text{argmin}\}$.

By chaining basic selection policies according to $f(\mathbf{A}) \circ g(\mathbf{A}) = f(g(\mathbf{A}))$, custom selection policies can be defined in the ontology. The following example illustrates the policy to identify all transport providers with maximum transport capacity in a composite agent:

$$\begin{aligned} sel_{TransportProvider}(A) = \\ & random\_sel(\mathbf{A}) \\ & \circ prop\_sel(\mathbf{A}, argmax, tcap) \\ & \circ size\_sel(\mathbf{A}, =, 1) \\ & \circ func\_sel(\mathbf{A}, TransportProvider) \\ & \circ \mathcal{P}^A \end{aligned}$$

where $\mathcal{P}^A$ is the powerset of all atomic agents in A and $random\_sel$ a tie-breaker function

$$random\_sel(\mathbf{A}) = \begin{cases} \emptyset, & \text{if } \mathbf{A} = \emptyset \\ \text{randomly picked} & \text{otherwise} \\ \text{element from } \mathbf{A}, & \end{cases}.$$

The inverse selection policy is denoted by $\neg sel(A) = A \setminus sel(A)$.

b) Composition operator: A composition operator $c(A, np, op)$ combines the numeric property values of all atomic agents that form an agent. Note that composition operators currently need to be hard-coded into the model. The default supported operator is defined as

$$c(A, np, +) = \sum_{a \in A} pvalue(\{\hat{a}\}, np)$$

where np is a numeric property.

c) Inference rule: Both composition operators and selection policies can be combined to form an inference rule for composite agents, e.g., for all properties relating to locomotion, such as nominal velocity:

$$pvalue(\widehat{A}, v_{\text{nom}}) = c(sel_{\,TransportProvider}(i(\widehat{A})), v_{\text{nom}}, +)$$

which maps to the v_{nom} property value of the only available *TransportProvider* or 0 if there is none.

More complex inference is required to compute the (remaining) transport capacity

$$pvalue(\widehat{A}, tcap) = c(sel_{\,TransportProvider}(i(\widehat{A})), tcap, +) - c(\neg sel_{\,TransportProvider}(i(\widehat{A})), tcon, +)$$

where $i(\widehat{A})$ represents the reverse mapping from type to agent. Inference rules are defined in the ontology, so that users can add their own rules.

C. Special Agent Type Properties

Some special agent properties exist, where in contrast to the generic agent type properties, the reasoning mechanisms are hard-coded into the model.

1) Safety: In MoreOrg, the computation of safety of an agent is a special property motivated through the space application context. Safety is based on resource redundancy under the assumption of possible component substitution (see Section II-A). The measure for redundancy is the central part of our safety heuristic, and it is based on the standard modeling approach for parallel and serial component-based systems (see [32]). Each resource can be associated with a probability of survival, so that an overall probability of survival can be computed using a function decomposition tree approach. Information about the probability of survival of components should be derived from an initial system identification and is ideally updated with performance and degradation information from the real system.

The reliability R_f, also referred to as probability of survival, of a single functionality f can be computed by accounting for parallel components, i.e., resources that are not strictly required but which can serve as replacement

$$R_f(t) = \begin{cases} 1 - \prod_{i=1}^{n}(1 - p_i(t)), & \text{parallel system} \\ \prod_{i=1}^{n} p_i(t), & \text{serial system} \end{cases} \quad (6)$$

where $p_i(t)$ is the time-dependent probability of survival with $0 \leq p_i(t) \leq 1$. While component degrading can be one reason for a change of the probability of survival, MoreOrg leaves the use of time dependence as future improvement and instead uses a static probability of survival with $t = 0$.

Definition (Functional reliability): $R(\widehat{A}, \mathcal{F})$ denotes the *reliability* of a set of required functionalities $\mathcal{F}$, which is provided by an agent $\widehat{A}$.

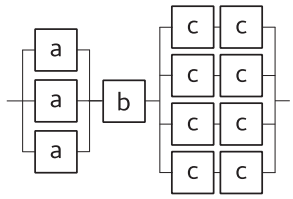


Fig. 4. Schematic of a system composition consisting of three resource types a, b, and c, where the ratio from required to available is 1:3 for a, 1:1 for b, and 2:8 for c.

The computation of $R(\widehat{A}, \mathcal{F})$ is based on the functional decomposition of the agent type $\widehat{A}$ into atomic resources. For each resource, a redundancy at component level (cf. [32]) is assumed. As a heuristic, the redundancy is computed based on a type partitioning considering all resources that have no further dependencies. All resources of the same type are modeled as subsystems, which again form a serial system. Fig. 4 illustrates this modeling approach.

For each subsystem, which is composed of a single resource type, the redundancy is computed for r required instances, n available instances, and the probability of survival p for the resource type:

$$rsub(r, n, p) = \begin{cases} 1 - [1 - p^r]^{\frac{n}{r}}, & \text{where } n \geq r \\ 0, & \text{otherwise} \end{cases}. \quad (7)$$

The function req maps a set of functionalities $\mathcal{F}$ to the required number of instances for each resource type

$$req(\mathcal{F}) = \{req_1, \ldots, req_{|req(\mathcal{F})|}\} \quad (8)$$

where req_i represents the minimum cardinality of a resource type i to fulfill all functionalities in $\mathcal{F}$.

The function avl maps an agent type to the number of maximum available resources with respect to a functionality set $\mathcal{F}$. Only resources that can contribute to the provision of $\mathcal{F}$ need to be considered

$$avl(\widehat{A}, \mathcal{F}) = \{avl_1, \ldots, avl_{|req(\mathcal{F})|}\} \quad (9)$$

where avl_i represents the maximum cardinality of a resource type i available in the general agent type $\widehat{A}$.

Resources lead to a heuristic system structure, as shown in Fig. 4, using serial and parallel systems. Based on this structure, an agent's reliability with respect to a set of functionalities is defined as

$$R(\widehat{A}, \mathcal{F}) = \prod_{i=1}^{|req(\mathcal{F})|} rsub(req_i, avl_i, p_i) \quad (10)$$

where p_i represents the probability of survival for a resource type i.

1) Example: Before introducing this example, note that functionalities can depend upon other functionalities or resources. Hence, an availability of functionality f_1 might imply the availability of a functionality set $\mathcal{F}_1 = \{f_1, f_2, f_3\}$, where f_2 and

TABLE III
RESOURCES OF THE AGENT TYPE *SHERPATT*, WHICH ARE RELEVANT TO PROVISION THE FUNCTIONALITY *LOCATIONIMAGEPROVIDER*

Resource	req_i	avl_i	p_i
Localization	1	1	0.95
Locomotion	1	1	0.95
Mapping	1	1	0.95
PowerSource	1	1	0.95
Camera	1	2	0.95

f_3 are direct dependencies of f_1. In this example, a functionality *LocationImageProvider* depends on a functionality *ImageProvider* and a functionality *MoveTo*. The requirements for the *ImageProvider* are one of each resource: *Camera* and *PowerSource*. *MoveTo* requires one of each resource: *Localization*, *Locomotion*, *Mapping*, *PowerSource*.

Table III lists the cardinalities and probabilities of survival for an atomic agent type *SherpaTT* with relevant resources.

This agent type provides the functionality *LocationImageProvider* with a redundant camera system, and otherwise a series system. According to (10), the probability of survival for the functionality *LocationImageProvider* is

$$P = (1 - (1 - 0.95)^2) \times \ 0.95^4 \approx 0.81.$$

A composite agent that has an additional atomic agent *PayloadBattery* can increase the redundancy of the resource *PowerSource* by 1. This leads to an increase of the probability of survival for the functionality *LocationImageProvider*, since now two redundant subsystems exist

$$P = (1 - (1 - 0.95)^2)^2 \times \ 0.95^3 \approx 0.85.$$

2) Efficacy: Efficacy in the context of MoreOrg describes the ability of a reconfigurable multirobot system to provide a particular functionality. To measure an agent's efficacy, an objective has to be given, here as a set of required functionalities. The identification of efficacy leads only to a binary result: either the organization supports the given functionality or not.

The definition of support can be viewed as an analysis of the redundancy level of components with respect to a required set of functionalities. The bottom-up definition of functionality support eventually leads to the definition of an agent's efficacy

$$\mathit{efficacy}(\widehat{A}, \mathcal{F}) = \begin{cases} 1, & \mathit{support}(\widehat{A}, \mathcal{F}) \geq 1 \\ 0, & \text{otherwise} \end{cases}. \qquad (11)$$

This definition of efficacy illustrates our approach to use resource summation to actively exploit redundancies for the creation of functional systems. For efficacy, only a sufficient component count is relevant, while the computation of the safety objective will take into account any excess resources.

3) Operation Cost: Typically, robotic systems consume electrical energy, and MoreOrg additionally expects a definition of all agents' nominal power consumption. Since we do not assume a homogeneous set of robots and might also operate in varying coalitions, agents will have a varying power consumption. Therefore, operation time only is not an accurate cost measure. Instead, MoreOrg uses the total energy consumption of a reconfigurable multirobot system as cost measure.

Power consumption can vary over time with the type of activity, but MoreOrg assumes a constant power consumption of all operative atomic agents and leaves a more sophisticated estimation, e.g., based on a functionality-based power consumption model, as future enhancement. MoreOrg estimates the operation cost as total consumed energy for all agents based on the nominal power consumption

$$\mathit{ocost}(\widehat{A}, t) = \sum_{\hat{a} \in \widehat{A}} \gamma_{\widehat{A}}(\hat{a})(pw(\hat{a}) \cdot t).$$

D. Organization Properties

In contrast to atomic and composite agents, an organization property can differ in its structure, i.e., might change its coalition structures over time. Therefore, we describe here the coalition structure properties of a reconfigurable multirobot system, as well as the cost to change between coalition structures.

1) Coalition Structure Properties:

a) Goal-dependent reliability: The organizational structure intends to support activities of its member agents that help to achieve and maximize the shared organizational goals. Agents can operate in parallel at different physical locations. The objective for an active coalition structure $CS = \{GA_1, \ldots, GA_{|CS|}\}$ of an organization is, therefore, described by a corresponding set of functionality sets denoted $FS = \{\mathcal{F}_1, \ldots \mathcal{F}_{|CS|}\}$, and $a2f : CS \rightarrow FS$ allows us to map each operative agent GA_i to the required functionality set $\mathcal{F}_i$. The current redundancy of the organization is then the minimum achievable level of redundancy

$$R(CS, FS) = \min_{A \in CS} R(\widehat{A}, a2f(A)). \qquad (12)$$

Note that this computation is not used in the planning approach, but will be used in the future to identify coalition structures with critically unbalanced resource distribution.

b) Efficacy: The efficacy of an organization's current coalition structure is computed from all operative agents' efficacy as

$$\mathit{efficacy}(CS, \mathcal{F}) = \min_{A \in CS} \mathit{efficacy}(\widehat{A}, \mathcal{F}). \qquad (13)$$

Here, all operative agents in a coalition structure need to support the functionalities in $\mathcal{F}$.

Note that MoreOrg implements the optimal coalition structure generation algorithm developed by Rahwan *et al.* [33] to search for a coalition structure, where $\mathit{efficacy}(CS, \mathcal{F}) = 1$. This optimization approach is used, e.g., to validate the feasibility of a transition of a set of agents between two locations. Since only mobile agents can relocate, a coalition structure is required such that $\mathit{efficacy}(CS, \{MoveTo\}) = 1$.

2) Operation Cost: Reconfiguration of an organization contributes to operation cost, since transitions between coalition structures require time and energy. The time to transition from one coalition structure CS_i^A to another CS_j^A when $CS_i^A \neq CS_j^A$ is, therefore, estimated with a heuristic function. The heuristic first assumes basic cost for the number of atomic

agents, which are involved in the reconfiguration. Second, additional and significantly higher cost arise from the need to coordinate multiple agents to exchange atomic agents or to merge. Therefore, the reconfiguration cost function to form a single agent from an existing coalition structure takes into account the number of general agents, equal to the partitions of the coalition structure CS, and the total number of involved atomic agents

$$\rho(GA, CS) = t_a \cdot |CS| + t_b \cdot |GA| \tag{14}$$

where t_a and t_b are heuristic time constants. Robotic experiments are required to establish a realistic estimate for the magnitude of these parameters. Our real-world experiments give an indication for these parameters for small teams, so that a default setting of $t_a = 600\,s$ and $t_b = 100\,s$ applies. The values are estimates, which consider time for additional error handling. The overall reconfiguration cost to transition from a coalition structure CS_i^A to another CS_j^A is defined as

$$\rho(CS_i^A, CS_j^A) = \sum_{GA \in CS_j^A} \rho(GA, CS_i^A). \tag{15}$$

Note that the reconfiguration cost heuristic does not account for relocation cost. Instead, we assume that all agents taking part in a reconfiguration process operate in direct proximity. Thus, this heuristic penalizes an involvement of an increasing number of agents to extract a new one. The first term penalizes the total number of involved independent agents to form a new agent with; each additional agent increases communication and coordination cost, as well as the likelihood for failures. The second term accounts for the fact that only a selected set of atomic agents is involved in a reconfiguration—the fewer the better.

IV. Exploitation of Redundancies

To exploit reconfigurable multirobot systems, we use a constraint-based mission planning approach. Constraints allow us to define the essential characteristics of a potential solution, by letting a user specify task requirements. The collected constraints then define a partial-ordered plan, for which a suitable full solution has to be found by translating the mission requirements to an agent allocation and constellation problem. Agent allocation here means that the exact organization structure and its development over time throughout a mission is initially unknown. What is known apart from the given constraints, however, are the total available atomic agents, their associated resources, and the model to combine atomic agents to composite agents in order to support required functionalities.

We implemented the planner Temporal Planning for Reconfigurable Multirobot Systems (TemPl) [26] to address the constraint-based planning problem for reconfigurable multirobot systems. This planner uses MoreOrg as the organization model, but adds a temporal and spatial dimension to problem definitions.

Definition (*Spatiotemporal requirement*): A spatiotemporal requirement is represented as a spatiotemporally qualified expression s, which describes the functional requirements and agent instance requirements for a time interval and a location:

$$s = (\mathcal{F}, \widehat{A}_s)@(l, [t_s, t_e])$$

where $\mathcal{F}$ is a set of functionality constants, $\widehat{A}_s$ is the general agent type representing the required agent type cardinalities, $l \in L$ is a location variable, and $t_s, t_e \in T$ are temporal variables describing a temporal interval with the implicit constraint $t_s < t_e$.

Each spatiotemporal requirement represents a persistence constraint, i.e., the requirements have to hold throughout the time interval. The mission specification allows us to relate spatiotemporal requirements to the organization model, which defines agent types and functionalities along with the associated properties.

Definition (*Mission*): A robotic mission is a tuple $\mathcal{M} = \langle \widehat{A}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$, where the general agent type $\widehat{A}$ describes the available agent types, STR is a set of spatiotemporally qualified expressions, $\mathcal{X}$ is a set of constraints, $\mathcal{OM}$ represents the organization model (here MoreOrg as described in Section III), T is the set of timepoints, and L is the set of locations.

A. Mission Constraints

Constraints in $\mathcal{X}$ can refer to spatiotemporally qualified expressions, and the initial state of a mission is defined by the earliest timepoint and binds available agents to their starting depot. The earliest timepoint is $t_0 \in T$ and $\forall t \in T, t \neq t_0 : t > t_0$. Note that this planning approach requires neither a single starting location for all agents nor a single final destination. An example for a mission is given in Table VIII.

1) Temporal Constraints: Temporal constraints are listed in Table IV. They allow us to define the temporal relation between spatiotemporal constraints in a relative and absolute manner.

2) Model Constraints: TemPl implements a subset of feasible (meta-)constraints (see Table V). Model constraints set requirements for agent types and agent roles. They allow bounding the cardinality of agent types so that the combinatorial search problem can be limited according to a least-commitment principle. Equality constraints allow us to restrict agent routes partially or even completely. Requiring the minimum equality of a single agent type over the full mission defines the complete route for a single atomic agent of this type. Thereby, modeling constraints allow us to detail a mission. In general, constraints apply to the dimensions space, time, agent types, and roles.

3) Functionality and Property Constraints: Agents either comprise a functionality or do not. In effect, functionality is requested with a maximum cardinality of one, which makes the introduction of a maximum function constraint unnecessary. Note that this is a limitation of the current modeling approach. However, the property of an agent providing a particular functionality can be of importance, and the use of property constraints (see Table VI) allows us to narrow applicable agents.

TABLE IV
TEMPORAL CONSTRAINTS FOR A MISSION $\mathcal{M} = \langle \widehat{A}, STR, \mathcal{X}, \mathcal{OM}, T, L \rangle$

Name	Syntax	Description
temporal relation	$\langle t_n, REL, t_m \rangle$	t_n and t_m are qualitative timepoints and REL is the set of permitted relations, so that $REL \subseteq \{<, >, =\}$ [9]
min duration	$minDuration(t_n, t_m, d)$	sets a lower bound for the duration of a time interval: $t_n - t_m \geq d$, where t_n and t_m are two qualitative timepoints $d \in \mathbb{R}^+$; implies the qualitative relationship $t_n > t_m$
max duration	$maxDuration(t_n, t_m, d)$	sets an upper bound for the duration of a time interval: $t_n - t_m \leq d$, where t_n and t_m are two qualitative timepoints $d \in \mathbb{R}^+$; implies the qualitative relationship $t_n > t_m$

TABLE V
MODEL CONSTRAINTS, WHERE $S \subseteq STR$ AND $\widehat{A}_s$ REPRESENTS THE GENERAL AGENT TYPE REQUIREMENT OF $s \in S$

Name	Syntax	Description
min cardinality	$minCard(S, \hat{a}, c)$	Minimum cardinality constraint $\forall s \in S : \gamma_{\widehat{A}_s}(\hat{a}) \geq c$, where $c \geq 0$
max cardinality	$maxCard(S, \hat{a}, c)$	maximum cardinality constraint corresponding to $minCard$ so that $\forall s \in S : \gamma_{\widehat{A}_s} \leq c$, where $c \geq 0$
all distinct	$allDistinct(S, \hat{a})$	$\forall s \in S : \bigcap A_s^{\hat{a}} = \emptyset$
min distinct	$minDistinct(S, \hat{a}, n)$	$\forall s_i, s_j \in S, i \neq j : \left\| A_{s_i}^{\hat{a}} \right\| - \left\| A_{s_j}^{\hat{a}} \right\| \geq n$, where $n > 0$
max distinct	$maxDistinct(S, \hat{a}, n)$	the equivalent maximum constraint to $minDistinct$, so that $\forall s_i, s_j \in S, i \neq j : \left\| A_{s_i}^{\hat{a}} \right\| - \left\| A_{s_j}^{\hat{a}} \right\| \leq n$, where $n \geq 0$
min equal	$minEqual(S, A_r)$	minimum existence of the same agent roles so that $A_{eq} = \bigcap_{s \in S} r(A_s)$ and $A_r \subset A_{eq}$, where $A_r \subset r(A)$, A is the available agent pool for a mission, and A_s is the agent pool that fulfils $s \in S$
max equal	$maxEqual(S, A_r)$	maximum existence of the same agent roles so that $A_{eq} = \bigcap_{s \in S} r(A_s)$ and $A_{eq} \subset A_r$, where $A_r \subset r(A)$, A is the available agent pool for a mission, and A_s is the agent pool that fulfils $s \in S$
all equal	$allEqual(S, A_r)$	the constraint conjunction: $minEqual(S, A_r) \wedge maxEqual(S, A_r)$

TABLE VI
FUNCTIONALITY AND PROPERTY CONSTRAINTS

Name	Syntax	Description
min function	$minFunc(s, f)$	functionality f to be available at spatio-temporal requirement (str) $s \in STR$, so that $f \in \mathcal{F}^s$, where $\mathcal{F}$ represents the functionality requirements associated with s
min property	$minProp(s, f, p, n)$	constrain the numeric property p_f of a functionality f to be $p_f \geq n$, where $n \in \mathbb{R}$ and $minProp(s, f, p, n)$ implies that $minFunc(s, f)$ holds
max property	$maxProp(s, f, p, n)$	equivalent maximum property value constraint to $minProp(s, f, p, n)$, so that property $p_f \leq n$, $n \in \mathbb{R}$

B. Planning and Optimization

The (high-level) optimization problem for a given mission $\mathcal{M}$ can be stated as

$$\begin{aligned} &\underset{\mathcal{M}^*}{\text{minimize}} \quad cost(\mathcal{M}^*, \mathcal{M}) \\ &\text{subject to} \quad STR \text{ and } \mathcal{X}. \end{aligned}$$

where

$$\begin{aligned} STR &\quad \text{spatiotemporal requirements} \\ \mathcal{X} &\quad \text{mission constraints} \\ \mathcal{M}^* &\quad \text{solution to mission } \mathcal{M} \\ cost(\mathcal{M}^*, \mathcal{M}) &= \alpha E(\mathcal{M}^*) \\ &\quad + \beta SAT(\mathcal{M}^*, \mathcal{M}) \\ &\quad + \epsilon SAF(\mathcal{M}^*, \mathcal{M}). \end{aligned}$$

The multiobjective cost function is based on the heuristics listed in Table VII and can be evaluated as soon as a solution for a mission exists.

The three objectives are presented in the cost function: efficiency through the energy cost function, efficacy through checking the level of fulfillment, and safety as a redundancy dependent survival metric; for balancing, the parameters α, β, and ϵ can be used. Note that safety and fulfillment have a value range [0, 1], so that α should account for normalization to [0, 1] for the energy cost; α, therefore, comprises a factor E_{max}^{-1}, where E_{max} is the maximum energy cost, which is either the allowed one or is extracted from existing mission solutions. The latter is done for our evaluation in Section V.

The importance of operation cost is controlled via α; a higher value will lead a preference of missions, which require less energy. The time of an agent's operation depends upon estimated travel cost, time for the actual requested operation or task, and

TABLE VII
HEURISTIC COST COMPUTATION ON THE SOLUTION $\mathcal{M}^*$ FOR A MISSION $\mathcal{M}$ (ADAPTED FROM [26])

Name	Syntax	Description
distance	$d(a, \mathcal{M}^*)$	traveled distance of an agent a in $\mathcal{M}^*$
operation time	$op(a, \mathcal{M}^*)$	time horizon of the mission; any location change introduces a lower bound Δt_{min} for time intervals by assuming a traversal with the mobile agent's nominal velocity $v_{nom}(\hat{a})$, i.e., $\Delta t_{min} \geq \frac{d(a,\mathcal{M}^*)}{v_{nom}(\hat{a})}$
energy	$E(\mathcal{M}^*)$	$E(\mathcal{M}^*) = \sum_{a \in A} ocost(\hat{a}, op(a, \mathcal{M}^*))$ as overall consumed energy per mission, by summing the consumed energy per agent a to perform $\mathcal{M}^*$;
safety	$SAF(\mathcal{M}^*, \mathcal{M})$	$SAF(\mathcal{M}^*, \mathcal{M}) = \min_{s \in STR} R(A_s^*, \mathcal{F}_s)$, where R denotes the functional reliability function (see Section III-C) , $\mathcal{F}_s$ is the required functionality to satisfy s and A_s^* the available and assigned agent in mission solution $\mathcal{M}^*$.
fulfilment	$SAT(\mathcal{M}^*, \mathcal{M})$	$SAT(\mathcal{M}^*, \mathcal{M}) = \frac{1}{\lvert STR \rvert} \sum_{s \in STR} sat(s, \mathcal{M}^*)$ represents the ratio of fulfilled requirements, where $sat(s, \mathcal{M}^*) = \begin{cases} 0 & \text{if constraint } s \text{ is not satisfied in } \mathcal{M}^* \\ 1 & \text{if constraint } s \text{ is satisfied in } \mathcal{M}^* \end{cases}$

time for reconfiguration. Our approach extends an approach used by Wurm *et al.* [35] to estimate the cost based on the travel time between two locations. We suggest to use nominal speed v_{nom} as default property for atomic agents, so that based on this information, a duration estimate for location changes of mobile agents can be computed. As long as no better estimation or other routing constraints are available, the line-of-sight distance between two locations is still the basis for the cost computation.

Parameter β controls the penalty for missions that can only be partially fulfilled. Here, the heuristic assumes that each requirement is of equal importance. Future approaches should also take a priority into account.

The preference of safer operations and thus agents with higher redundancy is controlled via ϵ. In principle, a high negative value of ϵ leads to a preference for a solution with a single yet highly redundant agent that can solve the mission.

For the search for an optimal transition between coalition structures, it has to be considered that due to a high degree of redundancy, a safer coalition structure might lead to lower efficiency. Therefore, any optimization has to trade safety and efficiency against each other and can lead only to a Pareto-optimal solution.

C. Algorithm

To tackle the given optimization problem, one or more solutions for a given mission have to be identified. Due to the need for various possible mappings between required functionality and satisfying agents, the problem cannot be directly solved in a classical form, e.g., with linear programming. The problem needs to be transformed with the help of the MoreOrg, so that existing optimization approaches can be combined to perform local optimization, here combinatorial optimization, coalition structure optimization, and min-cost flow multicommodity flow via linear programming.

The basic algorithm that we are currently using is illustrated in Fig. 5. We will outline the general approach and refer to [25] and [26] for further details. The algorithm involves a candidate generation step, which accounts for major constraints but not all required ones. Only after candidate optimization and subsequent characterization, it will be known whether the suggested solution candidate is feasible, i.e., whether the resulting quantitative temporal network is consistent and whether required reconfigurations and location transitions can be performed. Since we also permit partial solutions, i.e., efficacy < 1, feasibility does not necessarily imply that all spatiotemporal requirements of a mission are satisfied, but only that the activities that are part of the suggested solution can be performed. Generating solution candidates with an efficacy < 1: 1) accounts for a future prioritization of requirements and 2) permits the identification of Pareto-optimal solutions with respect to safety, efficacy, and efficiency.

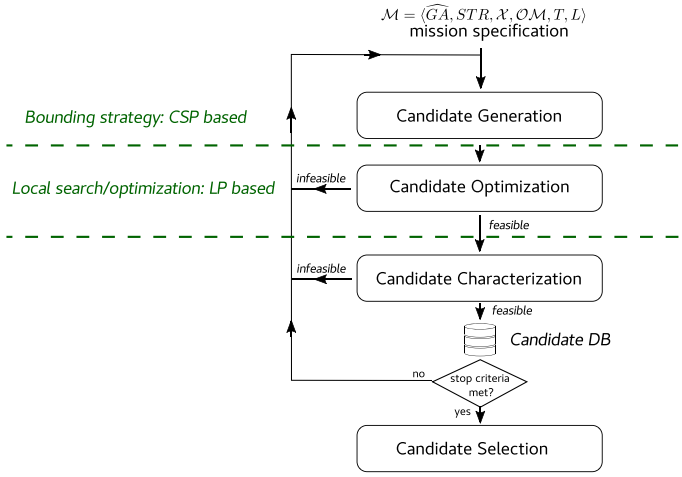


Fig. 5. Planning with Temporal Planning for Reconfigurable Systems (TemPl) uses a constraint-based solution candidate generation, where each candidate is characterized according to the mission objectives.

Initially, the planner aims at generating a resource efficient solution, by reducing the number of involved agents. It does

TABLE VIII
EXEMPLARY OUTLINE OF A SPACE EXPLORATION MISSION FOR A TEAM OF RECONFIGURABLE ROBOTS

$$
\begin{aligned}
\widehat{GA} =& \{(BaseCamp, 5), (CREX, 2), (CoyoteIII, 3), (Payload, 16), (SherpaTT, 3)\} \\
STR =& \{(\{(BaseCamp, 5), (CREX, 2), (CoyoteIII, 3), (Payload, 16), (SherpaTT, 3)\})@(lander, [t_0, t_1]), \\
& (\emptyset, \{(Payload, 3)\})@(lander, [t_5, t_{10}]), \\
& (\{LocationImageProvider, EmiPowerProvider\}, \{(Payload, 3)\})@(b_1, [t_2, t_3]), \\
& (\emptyset, \{(Payload, 1)\})@(b_1, [t_3, t_{14}]), \\
& (\{LogisticHubProvider, LocationImageProvider, EmiPowerProvider\}, \{(Payload, 3)\})@(b_2, [t_2, t_3]), \\
& (\emptyset, \{(BaseCamp, 1)\})@(b_1, [t_4, t_7]), (\{LocationImageProvider\}, \{(Payload, 3)\})@(b_4, [t_6, t_7]), \\
& (\emptyset, \{(Payload, 3)\})@(b_4, [t_8, t_9]), (\emptyset, \{(Payload, 1)\})@(b_6, [t_{10}, t_{14}]), (\emptyset, \{(Payload, 3)\})@(b_7, [t_{12}, t_{14}])\} \\
\mathcal{X} =& \{t_0 < t_1, \ldots, t_{13} < t_{14}\} \\
\mathcal{OM} =& \{\neg mobile(BaseCamp), mobile(CREX), tcap(CREX) = 2, mobile(CoyoteIII), tcap(CoyoteIII) = 4, \\
& \neg mobile(Payload), mobile(SherpaTT), tcap(SherpaTT) = 10, \ldots\} \\
T =& \{t_0, \ldots, t_{14}\} \\
L =& \{lander = (lat : -83.82009, long : 87.53932, moon), b_1 = (lat : -84.1812, long : 87.60494, moon), \\
& b_2 = (lat : -83.96893, long : 86.75471, moon), b_3 = (lat : -83.66856, long : 87.42557, moon), \\
& b_4 = (lat : -83.54570, long : 87.09851, moon), b_5 = (lat : -83.82009, long : 84.66000, moon), \\
& b_6 = (lat : -83.77371, long : 84.70960, moon), b_7 = (lat : -83.34083, long : 84.64467, moon)\}
\end{aligned}
$$

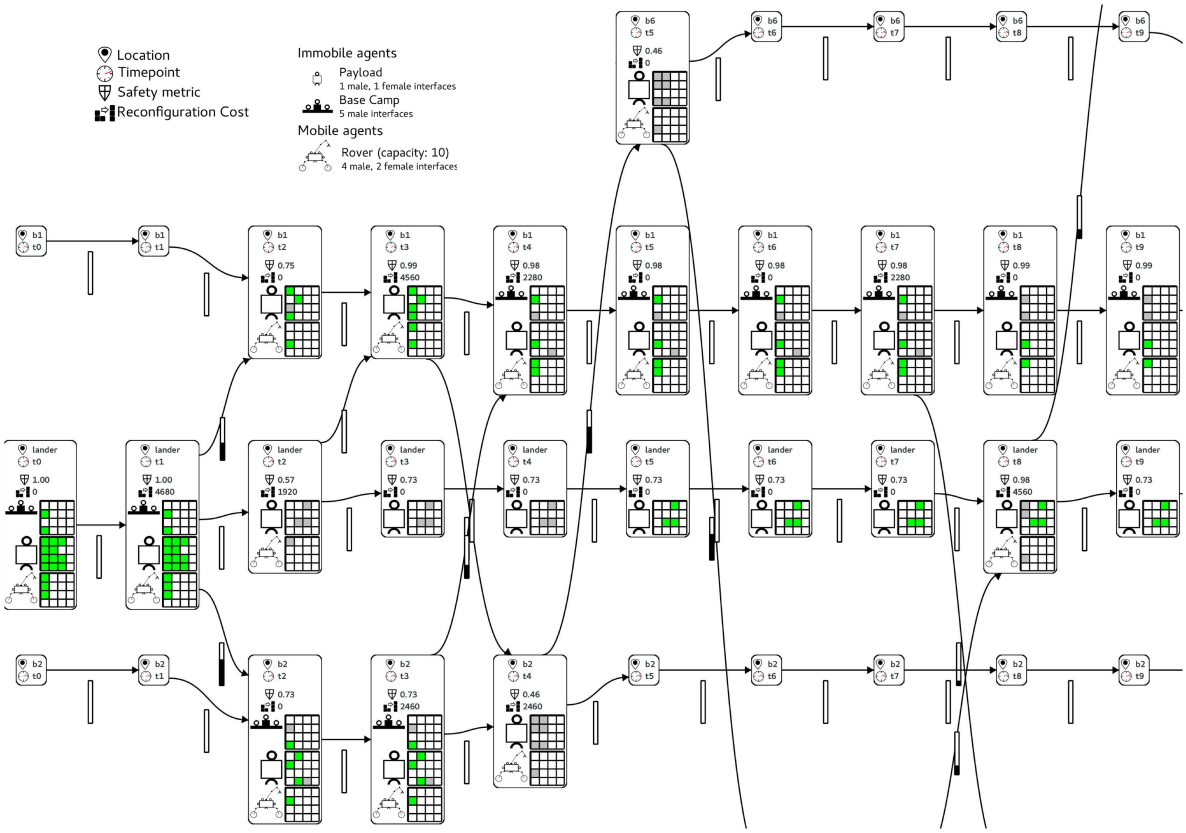


Fig. 6. Identified solution for the exemplary space mission after 20 min of search ($\psi_m = 2, \psi_{\neg m} = 0$) (to maintain readability only a part of the graph is placed here). Bars are annotating edges to illustrate transport capacity consumption, a green square identifies a required and available agent, while a gray square identifies an available though not required agent. Each constellation is attributed with the safety metric (probability of survival) and reconfiguration cost (in seconds).

so by bounding the number of agents that are assigned to mutually exclusive requirements through two parameters: ψ_m and $\psi_{\neg m}$. The default is $\psi_m = 0$ and $\psi_{\neg m} = 0$, which means that a set of mutually exclusive requirements only get the minimum needed resources to resolve their conflict: the parameters ψ_m and $\psi_{\neg m}$ refer to the number of additionally permitted mobile and immobile agents, respectively, to be assigned to mutually exclusive requirements. In effect, these parameters can increase the resource usage and level of redundancy for a solution and, at the same time, increase the options to find a solution that satisfies all requirements. A resulting increase of the number of agents can come, however, at a price: 1) higher complexity

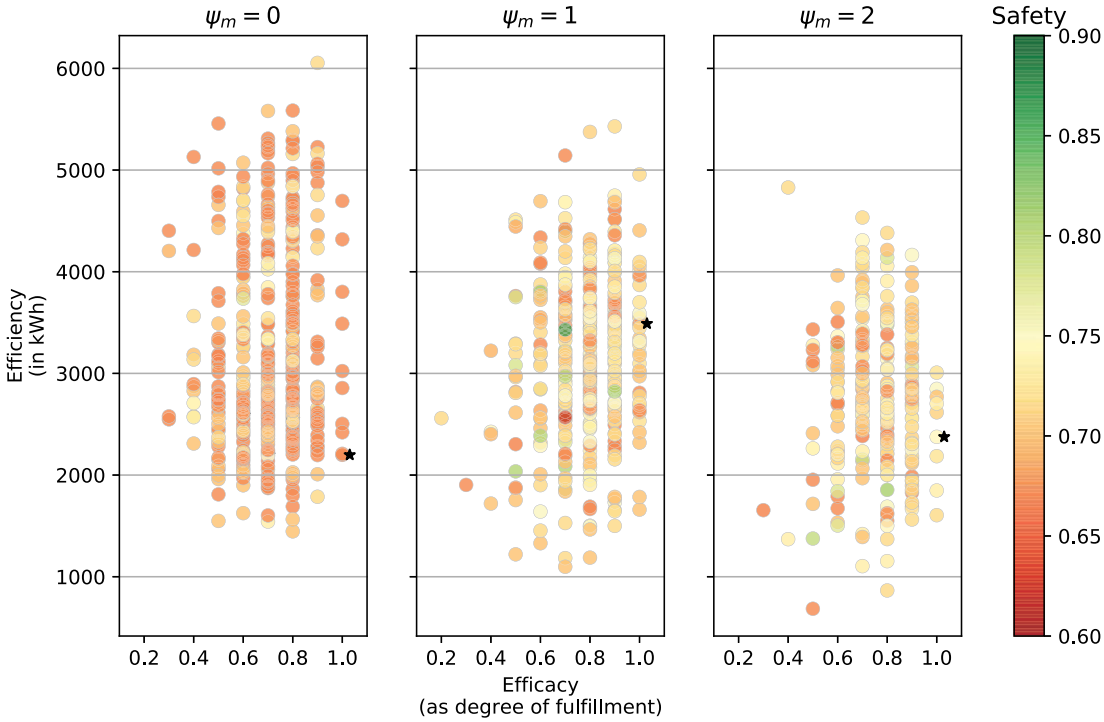


Fig. 7. Solution landscape compared for different bound settings. Black star-shaped markers identify the current local best solutions where the cost function is parameterized with $\alpha = 1.0$, $\beta = -100.0$, and $\epsilon = -10.0$. The results for each setting of ψ_m (efficiency, safety) are $\psi_m = 0$ (2198.25 kWh, 0.674), $\psi_m = 1$ (3489.5 kWh, 0.754), and $\psi_m = 2$ (2376.11 kWh, 0.747).

of the problem since additional agents allow for exponentially more reconfiguration options; and 2) higher operational cost, i.e., lower efficiency.

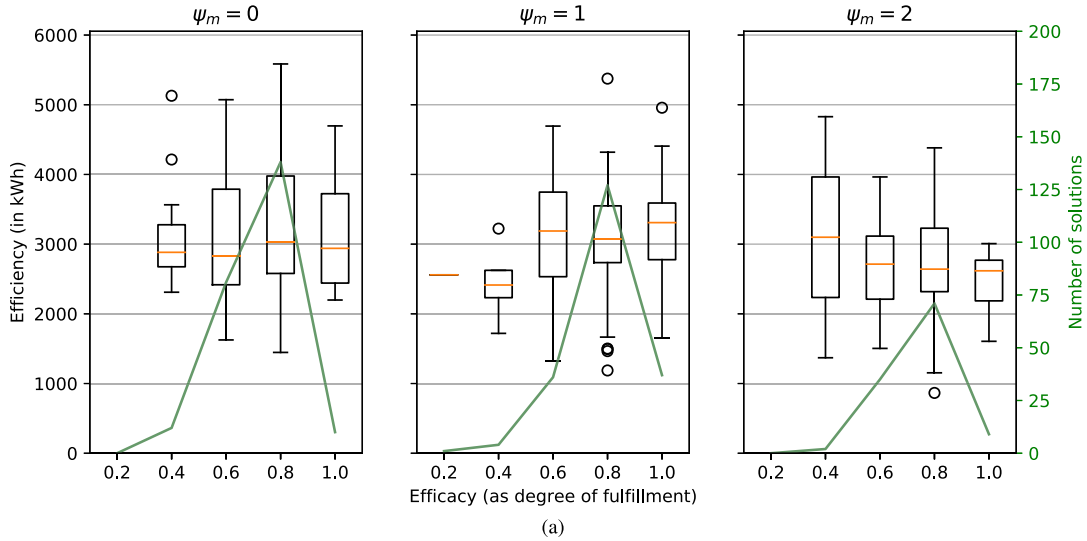
V. Evaluation

For an experimental evaluation of the planning approach, we look at a simulated space mission, which has been outlined by Sonsalla *et al.* [36]. The major goal of the mission is to place scientific payloads in a lunar environment at a predefined set of locations for science goals. These target science goals are cast into a respective mission specification, which is listed in Table VIII. Note that each location is defined by longitude and latitude in the specification; TemPl uses Mercator projection to convert these into Cartesian coordinates. A corresponding subsection of a solution found by TemPl is shown in Fig. 6. With respect to the safety measure, the starting assignment is ignored by assuming a probability of survival of 1.0. This is done to avoid an initial bias of the safety objective. To compute the safety objective, only relevant spatiotemporal requirements are analyzed. Hence, agents that are available at a space time point but are not actually required there (represented with gray colored boxes in Fig. 6) do not affect the safety objective (see, for instance, location $b6$, timepoint $t5$).

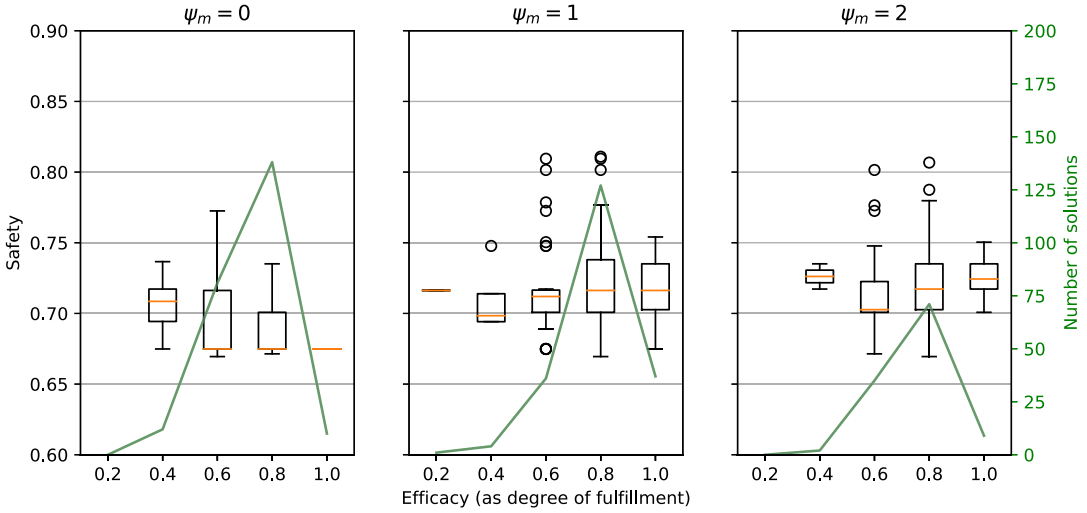
TemPl has been used to search for solutions to this mission scenario with a setting of $\psi_{\neg m} = 0$ and ψ_m in the range of 0–2. The search has been split into epochs with a maximum allowed planning time of 60 s. After 60 s, search has been reinitialized in order to escape from local minima. Planning has been stopped either after memory has been exhausted or when the total planning time of 20 min was exceeded (Intel 4th Gen i7-4600U 12-GB RAM). A higher setting of ψ_m requires additional agents to be used for the planning approach, so a higher computation time per solution is to be expected. Note that while the setting with $\psi_m = 0$ resulted in a stable range below 4 s per solution candidate, the required computation time per solution increases for $\psi_m = 2$ to up to 12 s—in combination with the total planning time, this explains the lower number of solutions found for higher settings of ψ_m. The comparison of solutions based on efficacy, efficiency, and safety is shown in Fig. 7 with additional details provided in Fig. 8.

What can be seen is that the expected increase in redundancy, due to the higher setting of the parameter ψ_m, leads to solution candidates with a consistently higher safety objective. This shows that ψ_m is an effective control parameter. By using an exemplary tradeoff setting for the cost function with $\alpha = -1$, $\beta = 100.0$, and $\epsilon = 10.0$, we can extract the solutions' characteristics. The black star-shaped marks indicate the best solutions according to the weights of the objective function. While safety can be increased from below 0.7 to approximately 0.75 for solutions with efficacy of 1, the efficiency deteriorates, although only slightly for $\psi_m = 2$ compared to $\psi_m = 0$.

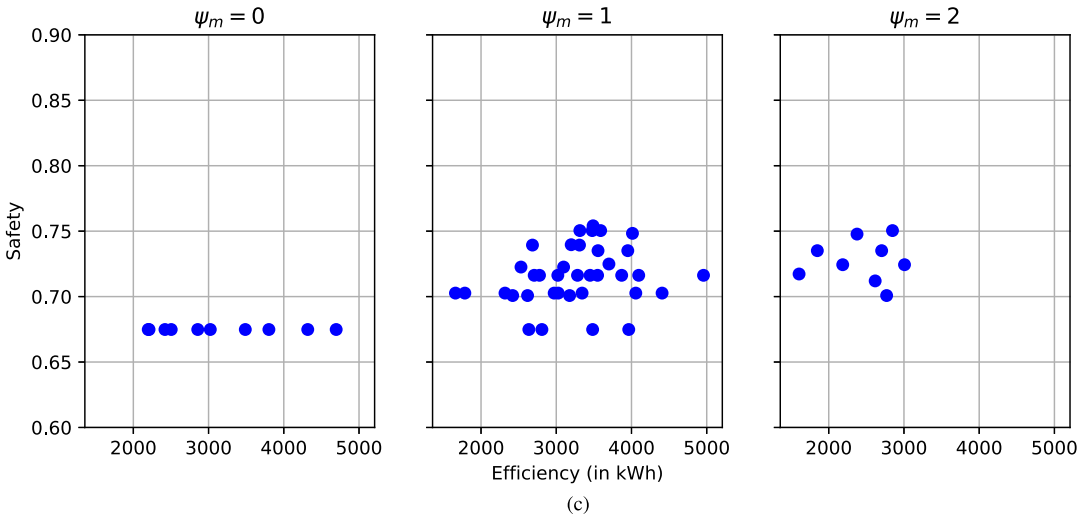
This analysis shows only an exemplary solution landscape, but at the same time the working of our modeling and planning approach for reconfigurable multirobot system capabilities. Our evaluation confirms that by increasing the options for agent assignments and at the same time the level of



(a)



(b)



(c)

Fig. 8. Analyzing the solution landscape of the space mission with respect to the optimization objectives. (a) Analyzing efficiency with respect to efficacy. (b) Analyzing safety with respect to efficacy. (c) Efficiency with respect to safety, where efficacy = 1.0.

redundancy of the reconfigurable system, improved solutions can be found for application scenarios, where safety might be an issue.

VI. Conclusion

This article outlined a modeling approach for reconfigurable multirobot systems, which permits an active exploitation of redundancies in multirobot systems through physical reconfiguration. The current approach does only moderately influence the level of redundancy in systems that are taking part in a robotic mission. This is mainly due to the complexity of the planning problem as a result of exponentially many reconfiguration options. It led us to use several heuristics and initially target resource efficient solutions. Clearly, this model might still not lead to solutions, which are sufficiently good from a safety perspective, but it offers a basis for an automated use of reconfigurable multirobot systems and an optimization strategy for organizational safety properties. Our deliberate planning approach can give an advantage compared to reactive reorganization strategies found in swarm-like systems by avoiding dead-end configurations. Meanwhile, a hybrid approach could be foreseen in a real application. Furthermore, we plan to augment already found solutions by explicitly routing available or rather unused agents along the critical path of a mission. This can be done without replanning until the known transport lines, which are established through mobile agents, are exhausted. Nevertheless, it comes again at the cost of efficiency.

The existing heuristics are based on assumptions and a limited set of practical experiments. As such, they can clearly be interpreted as a weakness of the modeling approach. The given heuristics shall, however, serve as basis and examples to develop better approaches to use reconfigurable multirobot systems. As such, they first point to the need/benefit for a submodel and second act as placeholders. All submodels and heuristics are subject of continued improvements and efforts to detail the model further. Probability of survival, for instance, as it is used and presented in this article, acts as a placeholder for more sophisticated metrics, e.g., some that are based on extensive empirical studies and specifications of subsystems. Furthermore, we plan to define safety not only through spatiotemporal requirements, but instead adapt the model to reflect the risks resulting from reconfiguration, idling, and relocation. Since a solution is characterized after all agents have been assigned, a dedicated model, e.g., for component degradation and reconfiguration errors, can be taken into account to improve the safety measure. We are also interested in using graph and network analysis techniques such as percolation [37] in combination with replanning to test the options to respond to system failures.

Reconfigurable multirobot systems combine the benefits of modular robots with capable robotic systems, and we see significant potential in developing further strategies and planning approaches. An automated exploitation will give designers of robotic missions a new degree of freedom. Still, significant practical challenges remain to exploit reconfiguration with real robots: increasing the reliability of all involved atomic agents is one challenge, and establishing reliable reconfiguration maneuvers is another. We do, however, outline here a feasible approach toward modeling and multiobjective planning for such systems.

Acknowledgment

The author would like to thank all contributors who helped to realize the multirobot team in the project TransTerrA.

References

[1] D. Brandt, D. J. Christensen, and H. H. Lund, "ATRON robots: Versatility from self-reconfigurable modules," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, 2007, pp. 26–32.

[2] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, and S. Murata, "Self-reconfigurable modular robot M-TRAN: Distributed control and communication," in *Proc. 1st Int. Conf. Robot Commun. Coordination*, 2007, Art. no. 21.

[3] J. Davey, N. Kwok, and M. Yim, "Emulating self-reconfigurable robots—Design of the SMORES system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 4464–4469.

[4] S. S. R. Chennareddy, A. Agrawal, and A. Karuppiah, "Modular self-reconfigurable robotic systems: A survey on hardware architectures," *J. Robot.*, vol. 2017, 2017, Art. no. 5013532.

[5] J. Liu, X. Zhang, and G. Hao, "Survey on research and development of reconfigurable modular robots," *Adv. Mech. Eng.*, vol. 8, no. 8, pp. 1–21, 2016.

[6] J. Baca, S. Hossain, P. Dasgupta, C. A. Nelson, and A. Dutta, "ModRED: Hardware design and reconfiguration planning for a high dexterity modular self-reconfigurable robot for extra-terrestrial exploration," *Robot. Auton. Syst.*, vol. 62, no. 7, pp. 1002–1015, Jul. 2014.

[7] F. Mondada, L. M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneubourg, and M. Dorigo, "The cooperation of swarm-bots," *IEEE Robot. Autom. Mag.*, vol. 12, no. 2, pp. 21–28, Jun. 2005.

[8] B. H. Wilcox *et al.*, "Athlete: A cargo handling and manipulation robot for the moon," *J. Field Robot.*, vol. 24, no. 5, pp. 421–434, 2007.

[9] P. Bartlett, D. Wettergreen, and W. R. L. Whittaker, "Design of the scarab rover for mobility and drilling in the lunar cold traps," in *Proc. 8th Int. Symp. Artif. Intell., Robot. Autom. Space*, Hollywood, LA, USA, 2008, pp. 1–8.

[10] W. Reid, R. Fitch, A. H. Göktoğan, and S. Sukkarieh, "Sampling-based hierarchical motion planning for a reconfigurable wheel-on-leg planetary analogue exploration rover," *J. Field Robot.*, vol. 37, no. 5, pp. 786–811, 2019.

[11] V. Dignum, Ed., *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Hershey, PA, USA: IGI Global, 2009.

[12] J. S. Evans, "Strategic flexibility for high technology manoeuvers: A conceptual framework," *J. Manage. Stud.*, vol. 28, no. 1, pp. 69–89, 1991.

[13] A. Zolli and A. M. Healy, *Resilience*. London, U.K.: Headline Publishing Group, 2012.

[14] J. F. Hübner, J. S. Sichman, and O. Boissier, "MOISE+: Towards a structural, functional, and deontic model for MAS organization," in *Proc. 1st Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2002, pp. 501–502.

[15] J. F. Hübner, J. S. Sichman, and O. Boissier, "Using the MOISE+ for a cooperative framework of MAS reorganisation," in *Proc. 17th Brazilian Symp. Artif. Intell.*, 2004, pp. 506–515.

[16] S. A. DeLoach, W. H. Oyenan, and E. T. Matson, "A capabilities-based model for adaptive organizations," *J. Auton. Agents Multiagent Syst.*, vol. 16, no. 1, pp. 13–56, 2008.

[17] S. A. DeLoach, "OMACS: A framework for adaptive, complex systems," in *Handbook of Research on Multi-Agent Systems*. Hershey, PA, USA: IGI Global, 2009, pp. 76–104.

[18] C. Zhong and S. A. DeLoach, "Runtime models for automatic reorganization of multi-robot systems," in *Proc. 6th Int. Symp. Softw. Eng. Adapt. Self-Manag. Syst.*, 2011, pp. 20–29.

[19] M. Tenorth and M. Beetz, "KnowRob: A knowledge processing infrastructure for cognition-enabled robots," *Int. J. Robot. Res.*, vol. 32, no. 5, pp. 566–590, 2013.

[20] M. Beetz, L. Mösenlechner, and M. Tenorth, "CRAM: A cognitive robot abstract machine for everyday manipulation in human environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 1012–1017.

[21] F. Cordes *et al.*, "LUNARES: Lunar crater exploration with heterogeneous multi robot systems," *Intell. Service Robot.*, vol. 4, no. 1, pp. 61–89, Jan. 2011.
[22] S. Bartsch, F. Cordes, S. Haase, S. Planthaber, and T. M. Roehr, "Performance evaluation of an heterogeneous multi-robot system for lunar crater exploration," in *Proc. 10th Int. Symp. Artif. Intell., Robot. Autom. Space*, 2010, pp. 30–37.
[23] T. M. Roehr, F. Cordes, and F. Kirchner, "Reconfigurable integrated multirobot exploration system (RIMRES): Heterogeneous modular reconfigurable robots for space exploration," *J. Field Robot.*, vol. 31, no. 1, pp. 3–34, Jan. 2014.
[24] R. Sonsalla *et al.*, "Field testing of a cooperative multi-robot sample return mission in mars analogue environment," in *Proc. 14th Symp. Adv. Space Technol. Robot. Autom.*, Leiden, The Netherlands, 2017, pp. 1–8.
[25] T. M. Roehr, "Autonomous operation of a reconfigurable multi-robot system for planetary space missions," Ph.D. dissertation, Dept. Math. Inform., Univ. Bremen, Bremen, Germany, 2019.
[26] T. M. Roehr, "Constraint-based mission planning for a reconfigurable multi-robot system," *Inteligencia Artif., Revista Iberoamericana de Inteligencia Artif.*, vol. 21, no. 62, pp. 25–39, 2018.
[27] T. M. Roehr and F. Kirchner, "Spatio-temporal planning for a reconfigurable multi-robot system," in *Proc. 4th Workshop Plan. Robot.*, 2016, pp. 135–146.
[28] G. Weiss, Ed., *Multiagent Systems*, 2nd ed. Cambridge, MA, USA: MIT Press, 2009.
[29] T. Shrot, Y. Aumann, and S. Kraus, "On agent types in coalition formation problems," in *Proc. 9th Int. Conf. Auton. Agents Multiagent Syst.*, 2010, pp. 757–764.
[30] S. Ueda, M. Kitaki, A. Iwasaki, and M. Yokoo, "Concise characteristic function representations in coalitional games based on agent types," in *Proc. Int. Joint Conf. Artif. Intell.*, 2011, pp. 393–399.
[31] D. Tsarkov and I. Horrocks, "FaCT++ description logic reasoner: System description," in *Automated Reasoning (ser. Lecture Notes in Computer Science)*, U. Furbach, and N. Shankar, Eds., vol. 4130. Berlin, Germany: Springer, 2006, pp. 292–297.
[32] M. Rausand and A. Høyland, *System Reliability Theory: Models and Statistical Methods*, D. J. Balding *et al.*, Eds., 2nd ed. Hoboken, NJ, USA: Wiley, 2009.
[33] T. Rahwan, S. D. Ramchurn, N. R. Jennings, and A. Giovannucci, "An anytime algorithm for optimal coalition structure generation," *J. Artif. Intell. Res.*, vol. 34, pp. 521–567, 2009
[34] R. Dechter, "Temporal constraint networks," in *Constraint Processing (The Morgan Kaufmann Series in Artificial Intelligence)*, R. Dechter, Ed. San Francisco, CA, USA: Morgan Kaufmann, 2003, ch. 12, pp. 333–362.
[35] K. M. Wurm, C. Dornhege, B. Nebel, W. Burgard, and C. Stachniss, "Coordinating heterogeneous teams of robots using temporal symbolic planning," *Auton. Robots*, vol. 34, no. 4, pp. 277–294, 2013.
[36] R. Sonsalla *et al.*, "Towards a heterogeneous modular robotic team in a logistic chain for extraterrestrial exploration," in *Proc. Int. Symp. Artif. Intell., Robot. Autom. Space*, 2014, pp. 1–8.
[37] M. Newman, *Networks*. Oxford, U.K.: Oxford Univ. Press, Mar. 2010.

Thomas M. Roehr was born in Duisburg, Germany, in 1981. He received the B.Sc. degree in engineering (information technology) from Baden-Wuerttemberg Cooperative State University, Stuttgart, Germany, in 2004, the M.Sc. degree in computer science from the Victoria University of Wellington, Wellington, New Zealand, in 2009, and the Ph.D. degree in department of mathematics and computer science from the University of Bremen, Bremen, Germany, in 2019 and in the area of Reconfigurable Multirobot Systems.

Since 2008, he has been a Researcher with the German Research Center for Artificial Intelligence—Robotics Innovation Center, DFKI GmbH, Bremen, Germany, where he leads the software backbone team. He has worked with reconfigurable multirobot systems over the past decade. He is also a maintainer of the open-source available Robot Construction Kit (Rock). His research interests include theoretical and practical aspects of the autonomous application of reconfigurable multirobot systems, and modular model-based software architectures for flexible and resilient robots.