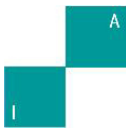
Inteligencia Artificial, 21(62) (2018), 25-39
doi: 10.4114/intartif.vol21iss62pp25-39



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

A Constraint-based Mission Planning Approach for Reconfigurable Multi-Robot Systems

Thomas M. Roehr
DFKI GmbH Robotics Innovation Center, Bremen, Germany
thomas.roehr@dfki.de

Abstract The application of reconfigurable multi-robot systems introduces additional degrees of freedom to design robotic missions compared to classical multi-robot systems. To allow for autonomous operation of such systems, planning approaches have to be investigated that cannot only cope with the combinatorial challenge arising from the increased flexibility of modular systems, but also exploit this flexibility to improve for example the safety of operation. While the problem originates from the domain of robotics it is of general nature and significantly intersects with operations research. This paper suggests a constraint-based mission planning approach, and presents a set of revised definitions for reconfigurable multi-robot systems including the representation of the planning problem using spatially and temporally qualified resource constraints. Planning is performed using a multi-stage approach, and a combined use of knowledge-based reasoning, constraint-based programming and integer linear programming. The paper concludes with the illustration of the solution of a planned example mission.

Keywords: Reconfigurable, Multi-Robot, Constraint-based, Vehicle Routing Problem

1 Introduction

Flexibility is the primary feature of reconfigurable multi-robot systems, since their modularity adds an additional degree of freedom to design robotic operations compared to the application of traditional multi-robot systems. For that reason Dignum et al. [11] discuss the so-called strategic flexibility, which offers an exploitation of proactive and reactive adjustment in the context of reconfigurable organizations. The strategic flexibility allows to tackle a set of unforeseen tasks with a robustly equipped system that allows recovery from malfunction thanks to increased redundancy. Exploiting strategic flexibility provides a strong motivation to combine increasingly capable autonomous robotic systems with a concept for modularity.

The main benefit of reconfigurable multi-robot systems lies in the fact that resources can easily, although not arbitrarily, be (re)used by any agent being a member of the reconfigurable system. Using this flexibility allows to balance resource usage and hence to adapt dynamically to operational demands. While modularity can lead to significant operational advantages it has drawbacks: if the level of modularity is chosen arbitrarily high this can lead to less capable systems. One can observe the effect in swarm-based systems, which come with a high degree of modularity: a swarm typically consists of cheap agents with a simple design, and thus , apart from emergent high-level behaviors, these agents come with a rather limited applicability by design. Although the mentioned emergent behaviors can be exploited, these behaviours remain harder to control or will be focused on a single task only. In general, reconfigurable multi-robot systems offer a feasible solution which consists of a mix of individually capable agents, including swarm-like units that can augment the overall robotic team. This augmentation is done either by acting as a fully autonomous agent or as an extension unit. Roehr et al. [17] implement this idea in the context of robotic space exploration missions in order to show the general feasibility and identify critical limitations: the

ISSN: 1137-3601 (print), 1988-3064 (on-line)

implemented approach validates the potential for increasing the flexibility in future robotic missions, but it also comes with increased operational demands. Thus, they suggest the introduction of a dedicated system model in order to automate operation of reconfigurable multi-robot systems and exploit the offered system capabilities to improve not only efficiency, but also safety of future robotic missions. Roehr and Kirchner [19] show how planning as essential element for automated operation for such a reconfigurable multi-robot system can be approached.

This paper details the problem definition and presents the results of the continued development of the planning approach. In Section 2 we briefly outline relevant background references for the state of the art. Section 3 introduces the planning problem, and in Section 4 we give details on the organization model and its extended use. Section 5 outlines the revised planning approach. We close with a conclusion and outlook in Section 6.

2 Background

The initial motivation for the planning problem is given by Sonsalla et al. [21], where a reconfigurable multi-robot system shall establish a logistics chain operation in order to support sample-return missions as part of extraterrestrial exploration. The robotic team consists of mobile and immobile agents, which can be physically connected via a set of electro-mechanical interfaces. By connecting one or more robots, they can form a new type of agent, comprising features none of the individual agents offers. The ability for reconfiguration offers novel ways of dealing with robotic missions, but Roehr and Kirchner [19] is to our best knowledge the only approach particularly dealing with reconfigurability. This mission planning problem can be understood as a logistic planning problem where mobile robots can transport other immobile and mobile robots. Hence, it is closely related to the Vehicle Routing Problem (VRP) [22]: a fleet of (most often homogeneous) mobile vehicles shall serve a set of customers, e.g., by delivering and/or picking up items, while minimizing a cost function – typically the overall travelled distance. The VRP applies to transportation and logistics scenarios and comes in many variants among which Capacitated VRP (CVRP), VRP with Time Windows (VRPTW) and VRP with Pick-up and Delivery (VRPPD) are the most popular ones. The pickup-and delivery problem can be further distinguished into a many-to-many (M-M), one-to-many-to-one (1-M-1), and one-to-one (1-1) problems, where the notation can be read as cardinalities for the origin, transition point, and target of a commodity, i.e. from-to or from-via-to. The M-M variant for example accounts for multiple commodity (good) origins and destinations, while the 1-M-1 variants assume a start and end of all vehicles at a single depot. The majority of these approaches are either focusing on a single-commodity case, homogeneous vehicle capacities or optimization of routing cost, were our approach has to deal with multi-commodities, heterogeneous vehicles, multi-depots, and fleet size optimization. Hence, a closer relation can be established to more specialized VRP approaches, e.g., such as the Heterogeneous or mixed Fleet VRP (HFVRP) [5] which accounts for a heterogeneous fleet and optionally with unlimited vehicle availability, or Dondo et al. [12] who approach Multi-depot heterogeneous fleet VRP with time windows (MDHFVRPTW). The variant VRP with Trailers and Transshipments (VRPTT) and more generalized VRP with multiple synchronization constraints (VRPMSs) [13] adds synchronization constraints between vehicles, which form a special instance of a reconfigurable multi-agent system containing agents or in this case vehicles of different categories: autonomous and non-autonomous, as well as support and task vehicles. Drexl [13] formulates a graph-based modelling approach to account for the interdependence of vehicles. He does not, however, provide an implementation of a solution approach.

While much of the research in VRP originates from the area of operational research, Coltin and Veloso [6, 7, 8] investigate a pick-up and delivery variant in the context of multi-robot systems and also apply their approach to a taxi problem with ridesharing. They implement optimal approaches as well as meta-heuristics, in particular simulated annealing, and Very Large Neighborhood Search (VLNS) [3] their application of VLNS results not only in a scalable approach, but also proves a general benefit of using transfers in a pickup and delivery scenario. In Section 3 we will outline the distinction between existing VRP and our approach, and provide additional constraints for our mission planning problem.

3 Mission Representation

The planning problem presented in the following aims to solve the problem of planning and scheduling a mission performed by a reconfigurable multi-robot system. While a mission can initially be seen as a task assignment for a multi-robot system, here it comes with an essential difference: agents are able to dynamically form physical coalitions referred to as composite agents. These composite agents are formed for three main reasons. Firstly,

to perform agent transport: one carrier agent attaches one or multiple (most likely, but not necessarily) immobile systems. Secondly, to provide functionality: some functionality is only available as so-called super-additive effect and requires two or more agents to join so that this functionality becomes available only for this composite agent, but not for the individual agents. Thirdly, to increase the functional redundancy: for agents that are assigned to fulfil requirements, we assume that adding relevant resources improves the redundancy and safety of operation, and effectively the likelihood of a successful performance of an agent.

While most VRP assume homogeneous agents, the team of agents in a reconfigurable multi-robot system is formed by heterogeneous agents; agents with individual capabilities and functionalities, as well as limitations to reconfigure and constraining attributes such as an overall transport capacity. We will look at a mission as a particular (minimal) partitioning problem of an agent team to achieve a requested agent- and function-distribution over space and time.

3.1 Definitions & Assumptions

In the following we introduce the basic notation, definitions and assumptions regarding reconfigurable multi-robot systems. The provided definitions describe a modular multi-agent system, which can form composite agents from a set of available agents:

Definition 3.1. *An **atomic agent** a represents a monolithic physical robotic system, where $A = \{a_1, \ldots, |A|\}$, is the set of all atomic agents, and $a \in A$ or equivalently $\{a\} \subseteq A$.*

Definition 3.2. *A mechanically coupled system of two or more atomic agents is denoted a **composite agent** CA, where $CA \subseteq A$, and $|CA| > 1$.*

Definition 3.3. *The type of an atomic agent a is denoted $\widehat{a}$ and equivalently for a composite agent CA the type is denoted $\widehat{CA}$. The set of all agent types is denoted $\theta(A) = \{1, \ldots, |\theta(A)|\}$, with the corresponding type-partitioned sets $A^1, \ldots, A^{|\theta(A)|}$, where $A = A^1 \cup \cdots \cup A^{|\theta(A)|}$.*

Definition 3.4. *A (general) agent is denoted GA, where $GA \subseteq A$, and $GA \neq \emptyset$. A (general) agent represents the wrapping concept for atomic and composite agents, with the corresponding type-partitioned sets $GA^1, \ldots, GA^{|\theta(A)|}$, where $GA = GA^1 \cup \cdots \cup GA^{|\theta(A)|}$*

Definition 3.5. *A (general) agent type $\widehat{GA}$ will be represented as a function $\gamma_{\widehat{GA}} : \theta(A) \to \mathbb{N}_0$, which maps an atomic agent type $\widehat{a}$ to the cardinality $c_{\widehat{a}}$ of the type partition, such that $c_{\widehat{a}} = |GA^{\widehat{a}}|$. The set of all constructible general agent types from a set of atomic agents A is denoted $\theta(\widehat{A})$; it represents the collection of all general agent types that are found in the powerset of all agents $\mathcal{P}^A$.*

Note, that a general agent type can equivalently be represented as tuple set of agent type and type cardinality: $\widehat{GA} = \{(\widehat{a}_1, c_1), \ldots, (\widehat{a}_n, c_n)\}$, where $a_i \in A$ and $c_i = |GA^i|$. $\widehat{GA} \supseteq \widehat{GA'} \iff \forall (a_i, c_i) \in \widehat{GA}, (\widehat{a}_i, c'_i) \in \widehat{GA'} : c_i \geq c'_i$, where $i = 1 \ldots |A|$.

Definition 3.6. *A set of atomic agents A is denoted an **agent pool** and it can be represented by a general agent type $\widehat{GA}$, such that $\forall a \in A : \gamma_{\widehat{GA}}(\widehat{a}) = |A^{\widehat{a}}|$.*

Definition 3.7. *An **atomic agent role** $r^{\widehat{a}}$ represents an anonymous agent instance of an atomic agent type $\widehat{a}$.*

Definition 3.8. *A coalition structure of an agent set A is denoted CS^A and is represented by a set of disjunct general agents $CS^A = \{GA_0, \ldots, GA_n\}$, where $GA_0 \cup \cdots \cup GA_n = A$, and $\forall i, j \wedge i \neq j : GA_i \cap GA_j = \emptyset$.*

3.2 Assumptions

Our design of the organization model and planning system for a reconfigurable multi-robot system, which both will be detailed in the following section, is based on a set of assumptions to simplify the modelling approach.

Assumption 3.1. *Each atomic and composite agent can be mapped to a single agent type only.*

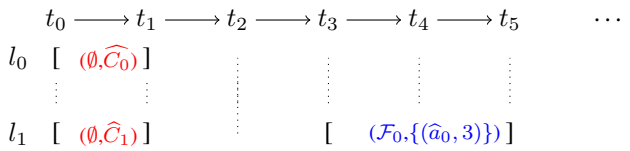


Figure 1: A mission specification example based on a space-time representation

A reconfigurable multi-robot system requires coupling interfaces, e.g., an electro-mechanical interface [10], to create physical linking between atomic agents to establish a composite system. Although multiple links could be considered between any two agents, interfaces cannot be arbitrarily coupled and the following assumption holds:

Assumption 3.2. *A mechanical coupling between two atomic agents can only be established through a single link and two and only two compatible physical coupling interfaces.*

3.3 Mission specification

The mission specification is a temporal database description, and it defines the initial, intermediate and goal state for a reconfigurable multi-robot system. A valid mission specification is described by the following two definitions:

Definition 3.9. *A spatio-temporal requirement is a spatio-temporally qualified expression (stqe) s which describes the functional requirements and agent instance requirements for a given time-interval and a particular location: $s = (\mathcal{F}, \widehat{GA}_r)@(l,[t_s,t_e])$, where $\mathcal{F}$ is a set of functionality constants, $\widehat{GA}_r$ is the general agent type representing the required atomic 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 $t_s < t_e$. Variables associated with s will also be referred to using the following notation: $\mathcal{F}^s$,$\widehat{GA}_r^s$,l^s,t_s^s, and t_e^s.*

Definition 3.10. *The robotic mission is a tuple $\mathcal{M} = \langle \widehat{GA}, STR, \mathcal{X}, \mathcal{OM} \rangle$, where the agent pool $\widehat{GA}$ describes the available set of agents, STR is a set of spatio-temporally qualified expressions, $\mathcal{X}$ is a set of constraints, and $\mathcal{OM}$ represents the organization model.*

The initial state is defined by the earliest timepoint and binds available agents to their starting depot. The earliest timepoint is $t_0 \in T$ and $\forall t \in T, t \neq t_0 : t > t_0$. Figure 1 illustrates a mission specification, where

$$\begin{aligned}
\widehat{GA} &= \{(\widehat{a}_0,3),(\widehat{a}_1,2)\},\\
STR &= \{(\emptyset,\widehat{C_0})@(l_0,[t_0,t_1]),(\emptyset,\widehat{C_1})@(l_1,[t_0,t_1]),\\
&\quad (\mathcal{F}_0,\{(\widehat{a}_0,3)\})@(l_1,[t_3,t_5])\}\\
\mathcal{X} &= \{t_0 < t_1,\ldots,t_4 < t_5\}\\
\mathcal{OM} &= \{mobile(\widehat{a}_0),\neg mobile(\widehat{a}_1),\ldots\}
\end{aligned}$$

$\widehat{C}_0 = \{(\widehat{a}_0,2),(\widehat{a}_1,1)\}$, $\widehat{C}_1 = \{(\widehat{a}_0,1),(\widehat{a}_1,1)\}$, l_0, l_1 are location variables and $t_0,\ldots,t_5$ are timepoint variables. Two general agents $\widehat{C}_0$ and $\widehat{C}_1$ are assigned to location l_0 and l_1 respectively. Two stqes related to the interval $[t_0,t_1]$ define the initial agent assignments; no functional requirements are part of the initial state description. The goal state is defined over the interval $[t_3,t_5]$ and requires a functionality set $\mathcal{F}_0$ in combination of least 3 agents of type $\widehat{a}_0$ at location l_1.

3.4 Mission constraints

A mission can be detailed by constraints in the constraint set $\mathcal{X}$. The only initially required constraints are temporal ones to describe the starting state, e.g., in the presented example all stqes relating to a start at t_0, e.g., cardinality constraints allow to set upper and lower bounds on the usage of agents and functionalities to reduce the combinatorial challenge. Other optional constraints can be added to detail and constrain the evolution of a mission. The following list describes the available constraint types; minimum constraints come with a corresponding max constraint implementation:

temporal qualitative timepoints describe time intervals, where timepoint constraints are provided using point algebra ($<, >, =$) [9].

duration $minDuration(s,t)$, $s \in STR$: sets a lower bound of time t for the duration of the time interval associated with the stqe s.

min cardinality $minCard(s, \widehat{a}, c_{min})$, $s \in STR$: represents a minimum cardinality constraint so that $|GA^{\widehat{a}}| \geq c_{min}$

all distinct $allDistinct(S, \widehat{a})$ describes the constraint: $\forall s \in S : \bigcap A^{\widehat{a},s} = \emptyset$, where $S \subseteq STR$, and $A^{\widehat{a},s}$ represents the subset of agents of type $\widehat{a}$ which are associated with the stqe s.

min distinct $minDistinct(S, \widehat{a}, n)$ describes the constraint: $\forall s_i, s_j \in S, i \neq j : \left||A^{\widehat{a},s_i}| - |A^{\widehat{a},s_j}|\right| \geq n$, where $n \geq 0$, $S \subseteq STR$, and $A^{\widehat{a},s}$ represents the partition of A which contains only agents of type $\widehat{a}$ which are associated with the stqe s.

all equal $allEqual(S, A_e)$ describes the constraint: $\forall s \in S \exists A_e : A_e = A_r^s$, where $A_e \subseteq A$, $S \subseteq STR$.

min equal $minEqual(S, A_e)$ describes the constraint: $\forall s \in S \exists A_e : A_e \subset A_e^s$, where $A_e \subset A$, $S \subset STR$.

min-function $minFunc(s, f)$: requirement for a functionality f to be available at stqe s: $f \in \mathcal{F}^s$

min-property $minProp(s, f, p, n)$ constrains the property p_f of a functionality f to be $p_f \geq n$, where the constraint implies $minFunc(s, f)$

To handle service preferences within this representation, e.g., when a particular agent should visit two distinct locations, equality constraints are required. An equality constraints can define partial or full paths for the same instances of agents, e.g., to control that the same agent visiting location l_0 at timepoint t_0 will also visit location l_1 at t_1. Detailing functionality request with min and max property constraints are motivated by informed repair strategies, e.g., a property constraint can demand a mobile agent with a particular transport capacity. In Section 4 we will detail this reasoning further.

3.5 Distinction & Observation

Existing VRP based approaches most often only consider a subset of the presented constraints, while the mission planning problem formulation embeds the following VRP properties: time windows, capacity constraints, heterogeneous agents, fleet size minimization and vehicle synchronization. Furthermore, additional special features are introduced: (i) it is not only accounted for commodity demand, but rather a combination of commodities and vehicles that provide certain functional properties; (ii) the use of qualitative temporal constraints (in contrast to hard or soft quantitative time windows), which enables partially ordered requirements and increase the flexibility to synchronize agent activities; (iii) the mix-in of a multi-pickup multi-delivery problem in contrast to a single drop-off.

4 Organization Modeling

To reason upon a reconfigurable multi-robot system a special so-called organization model is introduced which describes all resources that can be part of a reconfigurable multi-robot team: atomic agents as well as their functionalities and properties thereof. As detailed in [19] the organization model builds upon an ontological description, which: (a) encodes information about resources that are associated with agent types, (b) associates interfaces with agent types, (c) defines compatibility between interfaces, (d) allows the identification of feasible, and (e) allows inferencing functionality of composite agents.

In combination of all features the organization model serves as main reasoner to identify composite agents and coalition structures, which are suitable to support a set of time and location bounded functional requirements.

The following sections will describe agent properties, and the details of identifying feasible composite agents, and subsequently suitable agent with respect to a given functionality.

4.1 Atomic agent type

Each agent type is associated with the following essential attributes:

mobility $mobile(\widehat{a})$ defines whether an agent of type $\widehat{a}$ is mobile or not.

transport capacity $tcap(\widehat{a})$ defines the maximum total capacity (measured in storage units) of an agent of type $\widehat{a}$ to transport others, and $tcap(\widehat{a_i}, \widehat{a_j})$ defines the maximum capacity of an agent type $\widehat{a_i}$ to transport an agent type $\widehat{a_j}$.

capacity consumption $tcon(\widehat{a})$ defines the number of storage units an agent of type $\widehat{a}$ consumes temporarily when being transported (currently this is set to 1 by default);

velocity $v_{nom}(\widehat{a})$ defines the nominal velocity of an agent type $\widehat{a}$, $v_{nom} \geq 0$ for mobile atomic agent types and $v_{nom} = 0$ for immobile

power $pw(\widehat{a})$ defines the nominal required power to operate an agent of type $\widehat{a}$

mass $mass(\widehat{a})$ defines the mass of an agent

energy $energy(\widehat{a})$ defines the available electrical energy that initially comes with an atomic agent

4.2 General and composite agent type

Some properties of composite agents can be inferred from their compositing atomic agents: Avella et al. [4] (though in the context of route constraints) label these as 'numerical totalisable', e.g., here mass and energy, which can be easily represented as sum of the property values of each atomic agent forming the composite agent. Inferring the capacity, in contrast, can be complex due to geometrical packaging constraints. The present model, however, currently ignores geometrical packing constraints and checks only connectivity based on interface compatibility.

4.3 Feasible agents

The main feature of a reconfigurable multi-robot system is the possibility for physical interconnection, but the not all composite agents are feasible. The compatibility and availability of connecting interfaces can restrict the design of a fully connected composite agent. Interfaces can come in different variants, e.g., for the reference system in [18] a male and female (also referred to as *EmiPassive* and *EmiActive*). But only one male and one female interface can be coupled. Atomic agents can comprise any number of interfaces, but based on Assumption 3.2 exactly one interface can be used for the connection to another agent's interface. For a successful connection, both interfaces need to be compatible.

Checking feasibility is a matching problem for graph $G = (V, E)$, with constraints for the existence of edges, where a vertex $v \in V$ represents a single interface. We denote I^A as the set of all interfaces of a set of agent A, so that $V = I^A$ with the corresponding partitioning $I^A = I^0 \cup I^1 \cup \ldots I^n$, where $n = |A| - 1$ and the set of interfaces of an agent a_0 is represented as $I^0 = \{i_{0,0}, i_{0,1}, \ldots, i_{0,|I^0|-1}\}$. The adjacency matrix is an $m \times m$ Matrix C, where $m = |I^A|$, and $\forall i, j \in I^A : c_{i,j} = 0, 1$ (rows and columns are annotated with the interface):

$$
\begin{array}{c}
 \\
i_{0,0} \\
i_{0,1} \\
\vdots \\
i_{n,|I^n|}
\end{array}
\begin{array}{c}
\begin{array}{cccc}
i_{0,0} & i_{0,1} & \cdots & i_{n,|I^n|}
\end{array} \\
\begin{pmatrix}
c_{i_{0,0},i_{0,0}} & c_{i_{0,0},i_{0,1}} & \cdots & c_{i_{0,0},i_{n,|I^n|}} \\
c_{i_{0,1},i_{0,0}} & c_{i_{0,1},i_{0,1}} & \cdots & c_{i_{0,1},i_{n,|I^n|}} \\
\vdots & \vdots & \ddots & \vdots \\
c_{i_{n,|I^n|},i_{0,0}} & c_{i_{n,|I^n|},i_{0,0}} & \cdots & c_{i_{n,|I^n|},i_{n,|I^n|}}
\end{pmatrix}
\end{array}
$$

Checking connectivity means search for a valid assignment for the adjacency matrix C, while the following constraints hold for this symmetric matrix, where $c_{p,q} = c_{q,p}$, $p, q \in I^A$:

$$\forall a_k \in A, p, q \in I^k : c_{p,q} = 0 \tag{1}$$

$$\forall a_k \in A, p \in I^k : \sum_{q \in I^A} c_{p,q} \leq 1 \tag{2}$$

$$\forall a_k, a_l \in A : \sum_{p \in I^k} \sum_{q \in I^l} c_{p,q} \leq 1 \tag{3}$$

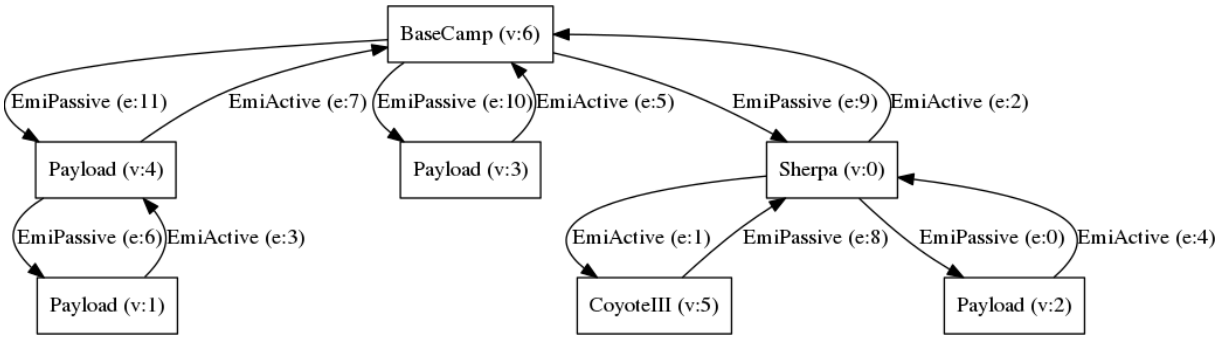


Figure 2: A feasible link structure for a composite agent after solving the assignment problem. Edges are annotated with the interface corresponding to the source vertex. Agent models and interfaces are related to the reference system described in [17].

Constraint 1 defines that no self links are allowed for an atomic agent, while Constraint 2 restricts each interface to be part of maximum one link only. Finally, Constraint 3 enforces Assumption 3.2, so that two atomic agents have to be connected by one link.

The assignment problem is solved using constraint-based programming and implemented using Generic constraint development environment (Gecode) [20], where the matrix entries represent the constraint-satisfaction problem (CSP) variables, each with the domain $D_c = \{0, 1\}$. Since a single agent might have multiple interfaces of the same type, the corresponding column assignments in the adjacency matrix are interchangeable, and create redundant solutions. We use symmetry breaking to reduce the number of redundant solutions, and to further speed the assignment process up, variable assignments are done in order of the least constrained agents, i.e.

$$a^* = \underset{a^k \in A}{\operatorname{argmin}} \frac{1}{|I^k|} \sum_{q \in I^A} \sum_{p \in I^k} c^*_{p,q} \tag{4}$$

, where

$$c^*_{p,q} = \begin{cases} 1 & \text{if } c_{p,q} \text{ is already assigned} \\ 0 & \text{otherwise} \end{cases}$$

In practice, we will also add a small fractional random bias which serves as tie breaker when between variables with equally constrained agents.

Figure 2 shows the result of a successful assignment procedure, for a set of seven agents, where the agent Sherpa comprises four male and two female interfaces, Payload one of each, CoyoteIII two male and BaseCamp five male.

4.4 Suitable agents

An atomic agent is associated with a set of resources, being either physical components or virtual ones such as capabilities and functionalities it can offer; the same holds for composite agents. Additionally, virtual resources can depend upon other resources, leading to a hierarchical dependency structure. In order to resolve the functional requirements of the mission specification to actual suitable agent type which support the requirements, the organization model provides a mapping function: $\mu : \mathcal{P}^{\mathcal{F}} \rightarrow \mathcal{P}^{\theta(\mathcal{A})}$, where $\mathcal{P}^{\mathcal{F}}$ represents the powerset of all functionalities, and $\mathcal{P}^{\theta(\widehat{A})}$ denotes the powerset of all general agent types. The function μ thus maps a set of functions to a set of general agent types which support this set of functions and forms feasible agents. The organization model encodes functionality based on resource availability, where resources can be physical devices and capabilities belonging to an agent. Thus, the organization model allows to infer functionality from a given agent type and its associated resource structure: $\mu^{-1} : \mathcal{P}^{\theta(\widehat{A})} \rightarrow \mathcal{P}^{\mathcal{F}}$. Thereby, the organization allows to map from agents to functionalities and back. An additional generalization can be achieved, when the mapping does not only account for a set of functionalities, but a set of arbitrary resource types which can be associated with a general agent. Currently, however, we have restricted the mapping to functionality.

Each agent type is associated with a maximum cardinality for a resource type, which reflects its initial and original state. Note, that setting the maximum cardinality still allows to lower the bound, in contrast to defining the exact cardinality. Therefore, the current modeling approach is prepared to consider resource failure or removal in future extensions.

Support is defined for an agent type and a single resource concept c as follows (cf. Roehr and Kirchner [19]):

$$support(\widehat{a}, c, f) \quad = \quad \frac{card_{max}(c, \widehat{a})}{card_{min}(c, f)} \tag{5}$$

, where $card_{min}$ and $card_{max}$ return the minimum and maximum required cardinality of resource instances. Accordingly, support of a function f with respect to a resource class c can be categorized as follows:

$$support(\widehat{a}, c, f) = \begin{cases} 0 & \text{no support} \\ \geq 1 & \text{full support} \\ > 0 \text{ and } < 1 & \text{partial support} \end{cases} \tag{6}$$

Since composite agents might comprise a high level of redundancy, the introduction of a saturation bound shall reduce the number of agents which have to be considered when a given set of functionalities is demanded. We define the *functional saturation bound* for an atomic agent type $\widehat{a}$ with respect to functionality f using the inverse of $support$:

$$FSB(\widehat{a}, f) = \max_{c \in \mathcal{C}} \frac{1}{support(\widehat{a}, c, f)}, \tag{7}$$

where $\mathcal{C}$ is a set of resource classes and $\forall c \in \mathcal{C} : card_{min}(c, f) \geq 1$ to account only for relevant resource classes. If there is no support for a $c \in \mathcal{C}$ such that $support(\widehat{a}, c, f)$ then $FSB(\widehat{a}, f) = \infty$. Similarly, the bound for a set of functions $\mathcal{F}$ is defined as:

$$FSB(\widehat{a}, \mathcal{F}) = \max_{f \in \mathcal{F}} FSB(\widehat{a}, f) \tag{8}$$

Identifying functionality support for a general agent type is equivalent to an atomic agent type, but to compute the maximum resource cardinalities the following holds:

$$card_{max}(c, \widehat{GA}) \quad = \quad \sum_{\widehat{a} \in \widehat{GA}} \gamma_{\widehat{GA}}(\widehat{a}) card_{max}(c, \widehat{a}) \tag{9}$$

, where $c \in \mathcal{C}$. Minimum resource cardinalities will be computed equivalently using $card_{min}(c, \widehat{a})$.

The number of general agent types that can support some functionality can be large, but it can be observed that for a supported set of functionalities a set of minimal general agent types $\mathcal{G}_{min}$ exists.

Definition 4.1. *A general agent type which supports a given set of functionalities and whose agent type cardinalities cannot be further reduced is denoted* ***minimal*** *with respect to the given set of functionalities.*

Hence, a minimal general agent type represents a lower bound to satisfy functionality requirements with a given combination of agent types.

5 Mission planning

The primary goal is to provide a valid assignment for the provided mission specification (cf. Section 3), while fleet size minimization and total cost minimization are secondary. The actual planning process is based on several stages in order to generate solutions:

(1) temporal ordering of all timepoints using a temporal constraint network
(2) upper and lower bounding of agent type cardinality for each spatio-temporal requirement
(3) generation of agent role timelines according to unification constraints and agent type cardinalities
(4) flow optimization to transfer immobile agents with mobile agents
(5) quantification of timepoints, based on transition times

Stage (1) creates a sequence of ordered timepoints, which is a necessary precondition for all next stages in order to identify concurrent resource usage and creating a commodity flow network. Stage (2) identifies the minimally required set of agent roles, and is for this reason a key element for minimization of the resources in use. Stage (3) takes all mission constraints into account in order to suggest a feasible agent role assignment. This assignment is the basis for local optimization in stage (4).

Constraint-based programming is involved in the reasoning of the organization model, and the planning stages (1),(2), and (3). Each of these stages involves the definition of appropriate branching strategies, and symmetry breaking conditions, and (3) uses of special implemented constraint propagator. If at any listed stage the search process fails, backtracking will be performed to the previous stage. The following sections will describe the details of the individual stages:

5.0.1 Temporal Ordering of Timepoints

To generate valid timelines and identify resource conflicts the approach requires a fully ordered set of timepoints. The generation of a fully constrained set of timepoints is based on qualitative temporal reasoning using point algebra with the set of relations $REL = \{>, <, =\}$ [16]. Consistency of the Temporal Constraint Network (TCN) is checked using a CSP which is defined by a set timepoint variables $T = \{t_1, t_2, \ldots, t_{|T|}\}$, a set $D = \{D_1, D_2, \ldots, D_{|T|}\}$ to represent the domain values for each timepoint $t \in T$, and a constraint set C with constraints of the form $C = \langle t_n, rel_i, t_m \rangle$, where $n, m = 1, \ldots, |T|$, and $rel_i \in REL$. A constraint is fulfilled if the relation described by $c \in C$ between two timepoint variables is fulfilled.

The final domain for each variable is restricted to a singleton: $|D_i| = 1$ and permitted values are $D_i \subseteq \{1, 2, \ldots, |T|\}$. If a full assignment of values can be found, the TCN is consistent, and the ordering of timepoints corresponds to the ordering of the assigned values. The qualitative temporal reasoning is sufficient to synchronize tasks, but only the quantification of time in the last stage of the planning approach will verify the temporal consistency of a solution.

5.0.2 Bounding agent type cardinality

To perform an upper and lower bounding of agent type cardinality a matrix based representation for spatio-temporal requirements and agent types is used, where $x_{i,j}$ represents the cardinality of agent type $\widehat{a}_j \in \widehat{A}$ and $s_i \in STR$. The following matrix representation with annotated rows and columns illustrates the meaning of each related CSP variable:

$$\begin{array}{c} \\ s_0 \\ s_1 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{cccc} \widehat{a}_0 & \widehat{a}_1 & \cdots & \widehat{a}_n \end{array} \\ \begin{pmatrix} x_{0,0} & x_{0,1} & \cdots & x_{0,n} \\ x_{1,0} & x_{1,1} & \cdots & x_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,0} & x_{m,1} & \cdots & x_{m,n} \end{pmatrix} \end{array} \tag{10}$$

, where $n = |\widehat{A}| - 1$, and $m = |STR| - 1$. Each variable x has an initial domain of positive integers $D_x = \{0, 1, \ldots\}$.

Since resource availability is restricted, the general agent type $\widehat{GA}$ which is part of the mission specification defines an upper bound for all agent type cardinalities, which will be referred to as $\widehat{GA}_{UB}$ for readability.

Spatio-temporal requirements, however, can overlap, i.e. when they refer to the same location and their time intervals overlap. For a set of overlapping spatio-temporal requirements $\Omega = \{s_i, \ldots, s_j\}$, $s_i, s_j \in STR$ the upper bound is enforced as follows:

$$\forall \widehat{a}_j \in \widehat{A} : \qquad \sum_{s_i \in \Omega} x_{i,j} \leq \gamma_{\widehat{GA}_{UB}}(\widehat{a}_j) \tag{11}$$

The organization model is required to translate the requirements for functionalities into requirements for (suitable) general agent types, and apply the functional saturation bound. Lower bounds for each spatio-temporal requirement result from the combination of demanded functionalities and the given minimum agent type cardinalities. This lower bound represents a set of minimal general agents which is translated into the spatio-temporal

requirement's CSP variable domain. This domain is considered in the CSP by using extensional constraints for the assignment of model cardinalities, thus restricting model combination to minimal general agents. The extensional constraints enforce an exact assignment, but any full assignment of model cardinalities is only a lower bound for the subsequent agent role assignment stage. If no assignment can be found, too few resources are available to fulfil the mission requirements; the planning continues with another assignment of the temporal constraint network if possible or fails otherwise.

5.0.3 Agent roles

Subsequent to the CSP branching on bounded agent type cardinalities, a candidate assignment of agent roles to spatio-temporal constraints can be computed using a set of integer variables $y_{i,k,j}$, for $s_i \in STR$, $\widehat{a}_k \in \widehat{A}$, and $0 \leq j \leq \gamma_{\widehat{GA}_{UB}}(\widehat{a}_k)$, which have the domain $D = \{0, 1\}$:

$$\begin{array}{c} \\ s_0 \\ s_1 \\ \vdots \\ s_m \end{array} \begin{array}{c} \begin{array}{cccccc} & r_0^{\widehat{a}_0} & \cdots & r_0^{\widehat{a}_k} & \cdots & r_l^{\widehat{a}_n} \end{array} \\ \begin{pmatrix} y_{0,0,0} & \cdots & y_{0,k,0} & \cdots & y_{0,n,l} \\ y_{1,0,0} & \cdots & y_{1,k,0} & \cdots & y_{1,n,l} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ y_{m,0,0} & \cdots & y_{m,k,0} & \cdots & y_{m,n,l} \end{pmatrix} \end{array} \tag{12}$$

, where $l = \gamma_{\widehat{GA}_{UB}}(\widehat{a_n}) - 1$, $m = |STR| - 1$, and $n = |\widehat{A}| - 1$.

Additional constraints are applied to guarantee unary agent role usage for time-overlapping constraints, and the general mission constraints described in Section 3 can directly be translated into low level CSP constraints, e.g., such as equality constraints minEqual, maxEqual as well as distinction constraints. Since agent roles of the same agent type are interchangeable symmetry breaking is applied to reduce the number of redundant solutions. While constraint propagation will reduce the corresponding domain and will lead to value assignment, full assignment of variables will only be performed for agent roles that (a) have an assignment apart from the single starting location, and (b) are mobile. To the first kind of agent roles we will also refer to as *active* agent roles. This partial assignment allows to extract full timelines for active mobile agents and partial timelines for active immobile agents. Both form the basis for a multi-commodity flow problem which is solved using integer linear programming.

5.0.4 Timeline Generation

Variable assignment for a single agent role variable assignment have to fulfill another important property: they have form a path in a temporal-expanded network. Ford and Fulkerson [14] have shown that networks can represent flow over time, and we similarly rely on what we call a temporal-expanded network to compute a flow-based representation for the mission planning problem. The temporal-expanded network has a bound on the number of edges by allowing only edges between vertices which are related to neighbouring timepoints and point forward in time:

Definition 5.1. *A time-expanded network for a set of timepoints T and a set of locations L is a graph $G = (V, E)$ with the following properties: Each vertex in V corresponds to a unique location timepoint tuple $v_{l,t} = (l, t)$, where $l \in L$, and $t \in T$. The set of edges is restricted: $e \in E \implies e = (v_{t_n,l_i}, v_{t_{n+1},l_j})$, where $n = 0, \ldots, |T| - 1$ and $i, j = 1, \ldots, |L|$. Without loss of generality $t_0 \leq t_1 \leq \cdots \leq t_{|T|-1}$.*

A custom (path) propagator has been implemented to exploit the structure of the network and enforce a constrained path in the network. This leads to a faster assignment process of agent role variables.

5.0.5 Multi-commodity flow

When the role assignment process is completed (fully for the mobile agents, and partially for the immobile ones), it is straightforward to translate the agent role timelines into a multi-commodity min-cost flow problem [2]: mobile agents represent transport providers, while immobile agents will be treated as individual commodities. Thus, edges in the network are either 'local' connections since they refer to the same location, or they are part of mobile agent routes. While we assume that local connections have infinite capacity, edges created as result of a mobile agent

transition have an upper capacity bound defined by the transport capacity of the corresponding mobile agent. All available mobile agents span a flow network over which commodities, or here immobile agents, can be routed to their target destinations. But agents are not restricted to a single target destination, so that requirements partially define a route for each agent. Therefore, the flow network represents all immobile agent requirements by minimum trans-flow requirements. Although bundling all agent types into one commodity would lead to a compact representation, route requirements for individual agents could not be set properly. Hence, each immobile agent role corresponds to a commodity, and role usage requirement are translated in to minimum transition requirements as already mentioned.

Mobile and immobile agent routes are transformed into a multi-commodity min-cost flow problem with unit commodity cost [2]:

$$\min \sum_{k,m} x_m^k$$

$$s.t. \sum_{e_m \in A_n} x_m^k - \sum_{e_m \in B_n} x_m^k = \begin{cases} S_k^+ & \text{if } n = s_k \\ -S_k^- & \text{if } n = t_k \\ 0 & \text{otherwise} \end{cases}, \forall\, n, k$$

$$x_m^k \geq l_m^k \wedge x_m^k \leq u_m^k$$

, where

$$\begin{aligned} G &= (V, E) \\ K &= \text{number of commodities}, k = \{1, \ldots, K\} \\ m &= \{1, \ldots, M\}, M = |E| \\ e_m &= \text{edge between node i and node j, i.e. (i,j)} \\ x_m^k &= \text{flow for commodity k in arc } e_m \\ c_m^k &= \text{unit cost for commodity k in arc } e_m \\ u_m^k, l_m^k &= \text{upper/lower bound for commodity k flow through edge m} \\ s_k, t_k &= \text{source/target of commodity k, } s_k \in V \\ S_k^+, S_k^- &= \text{supply/demand of } s_k \in V \\ B_n &= \text{set of incoming edges of node n} \\ A_n &= \text{set of outgoing edges of node n} \end{aligned}$$

To solve the network flow problem, the problem is first translated into a standard representation (CPLEX LP) so that different LP solvers can be used to solve the optimization problem (here: SCIP [1] and GLPK [15]). Any feasible and optimal solution of the network flow problem is also a feasible, but not necessarily an optimal solution for the mission assignment problem.

5.0.6 Quantification of time

A full solution still requires the quantification of temporal intervals: the qualified temporal network is therefore converted into a quantitative simple temporal network where the transitions between locations (and stqes) are based on the time required for the mobile systems to perform the location transitions and to form composite agents. Any min and max duration constraints will also apply at this planning stage.

5.1 Search & Solution repair

The previously described constraints lead to the generation of role timelines, and the CSP framework Gecode [20] has been used for the implementation. All role timelines are not only checked for feasibility via the multi-commodity min-cost flow optimization, but at the same time locally optimized. Still, finding a feasible solution for a highly restricted set of resources can be a significant challenge. Several strategies can be considering for search,

and our initial approach interprets lower agent type cardinality bounds as exact bounds - with the intention to keep the fleet size minimal and enlarge only when necessary. Hence, in the case when no optimal solution can be found, the infeasible (LP) solution is analysed to identify open flaws, i.e. unfulfilled minimum commodity trans-flow requirements. Upon identification of all flaws, a repair heuristic can be applied which injects additional transport provider requirements, thereby triggering either the change of existing mobile agent routes, or an increase of the lower agent type cardinality. The min-property constraint is used to augment the mission and restart the search after the local repair. For highly constrained missions, the repair process can reduce the number of flaws, but is slow at finding feasible solutions, hence showing that the heuristic is currently insufficient for complex setups.

An alternative is offered by the relaxation of cardinality bounds. In order to speed up finding an initial feasible solution, it is beneficial not to interpret the lower agent type cardinalities as exact bounds. Allowing an additional set of mobile systems (still within the number of the available ones) can reduce the time to find a feasible solution, but leads to higher redundancies and therefore less efficient solutions, since more agents will be required.

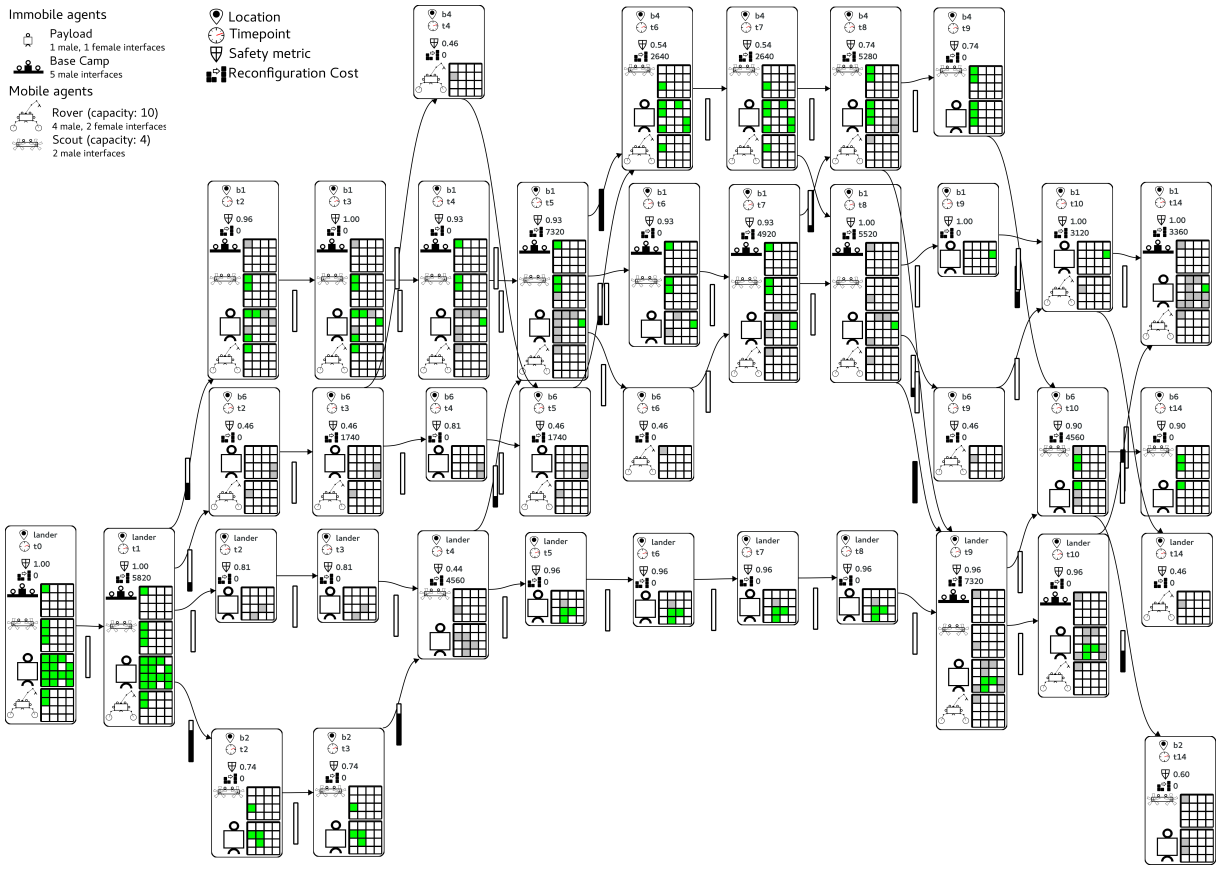


Figure 3: Example of a feasible solution for a full mission with locations = {lander,b1,b2,b4,b6}, timepoints = {t0,t1,...,t10,t14}. Fillbars indicate the consumed capacity of mobile agents. Color-coded boxes represent unique agent roles (only for better visualization limited to 16 per agent type): green = fulfilled requirement, gray = presence without a requirement.

Figure 3 shows a computed feasible solution. The general agents available for the mission are 3 Sherpa, 2 CREX, 3 Coyote II, 16 Payload, and 5 BaseCamp, where some agent interfaces are listed in the upper left corner of the figure. The assignment at the location lander shows, that only a subset of atomic agent is required for the solution. Fulfilled atomic agent requirements are highlighted as green squares, while the presence of systems without requirements is shown in green. These requirements, however, only represent one feasible set of atomic agent requirements which has been inferred from required functionalities. This solution has been computed within few seconds but only for a relaxed cardinality bound with two additional mobile agent roles (per mobile agent type).

5.2 Mission solution & cost function

The overall state of the agent organization, i.e. current connection state of atomic and composite agents is reflected by the coalition structure. In order to cost factor the dynamics in an agent organization two related concepts have to be used: policies and heuristics. Policies are required to define rules for selection and attribution. For example in the case of a transport multiple mobile robot may be available to perform this transport. To decide which one to take, a transport policy has been introduced, which chooses the agent with the largest transport capacity. For attribution energy consumption in a composite agent serves as main example. Since multiple power sources might exist is such system, a consumption policy has to distribute the consumption to all energy providers. Here, for the default policy each provider takes a share relative to its contribution to the overall energy capacity of the composite agent. Heuristics serve to interpolate a organizational state and estimate final mission costs: a duration heuristic for moving between locations relies on the information about the distance and the nominal speed of the transporting agent. Energy cost are depending upon the duration heuristic by relating duration to the power consumption of a composite system. Any reconfiguration changes this coalition structure, but requires a transition time, so that $\rho(CS_i^A, CS_j^A)$ defines the time to transition from one coalition structure CS_i^A to another CS_j^A. This cost heuristic assumes the same location of all agents in A.

The objectives of the planner is to find a solution that balances the overall energy consumed with the level of safety:

distance $d(a, \mathcal{M}_s)$ travelled distance of an agent a in mission $\mathcal{M}_s$

operation time $op(a, \mathcal{M}_s) = d(a, \mathcal{M}_s)/v_{nom}(a)$ duration of operation of an agent a

energy $E(a, \mathcal{M})$, where $E(a, \mathcal{M}_s) = op(a, \mathcal{M}_s) \cdot pw(\widehat{a})$ overall consumed energy by agent a to perform $\mathcal{M}_s$; $E(\mathcal{M}) = \sum_{a \in A}$ overall consumed energy per mission

safety $SAF(\mathcal{M}_s) = \min_{s \in STR} saf(s)$ represents the minimal safety level (here: redundancy) of the mission, where $saf(s)$ defines the safety metric associated with an stqe s based on the available (general) agent and with respect to the required set of resources; currently a redundancy based model is used to estimate the probability of survival based on an agent's set of component required to provide the functionalities in $\mathcal{F}$ (cf. [19] for details), such that $0 \leq saf(s) \leq 1$.

fulfillment $SAT(\mathcal{M}) = \frac{1}{|STR|} \sum_{s \in STR} sat(s)$ represents the ratio of fulfilled requirements, where

$$sat(s) = \begin{cases} 0 & \text{, } \textit{unfulfilled} \\ 1 & \text{, } \textit{fulfilled} \end{cases}$$

This following cost function reflects a balancing of three general mission aspects: efficiency through the energy cost function, efficacy through checking the level of fulfillment, and safety as redundancy dependant survival metric; for balancing the parameters α, β and $gamma$ can be used:

$$cost(\mathcal{M}_s) = \alpha E(\mathcal{M}_s) + \beta SAT(\mathcal{M}_s) + \gamma SAF(\mathcal{M}_s)$$

Figure 3 shows an example of a feasible solution. Each such solution can be translated into action plans for individual agent roles. Each vertex of the solution network serves as synchronization point and assumes reconfiguration operation to account for necessary coalition structure changes; the reconfiguration cost are annotated accordingly, along with the safety metric. Overall cost for the provided solution network are computed by constructing a simple temporal constraint network [9] where the bounds are defined by the transition times of the mobile agents.

6 Conclusion & Future Work

This paper presents the continued work for developing a planning system for a reconfigurable multi-robot system. The planner relies on constraint-based programming to specify and solve missions involving reconfigurable multi-robot systems, which is combined with multi-commodity flow optimization as local search. Furthermore, it suggests a multi-objective optimization target involving efficacy, efficiency and safety. The approach presented in this paper does not only result in a planning system for reconfigurable multi-robot system, but also in a tool which allows to analyse the effects of using reconfigurable multi-robot systems in robotics missions. Future work

will firstly focus on introducing better plan repair heuristics, and the extended use of meta-heuristic search strategies to improve the performance and scalability of the embedded local search approach. Secondly, a resource augmentation stage will be added in order to use previously unused resources to raise the level of safety.

Acknowledgments

This work was supported by the German Space Agency (DLR) under grant agreement 50RA1301 and 50RA1701, and the project Hi-Digit Pro 4.0.

References

[1] Tobias Achterberg et al. "Constraint Integer Programming: A New Approach to Integrate CP and MIP". In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Vol. 5015 LNCS. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 6–20. ISBN: 354068154X. DOI: 10.1007/978-3-540-68155-7_4.

[2] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Michigan, US: Prentice Hall, 1993, p. 846.

[3] Ravindra K. Ahuja, James B. Orlin, and Dushyant Sharma. "Very large-scale neighborhood search". In: *International Transactions in Operational Research* 7.4-5 (Sept. 2000), pp. 301–317. ISSN: 0969-6016. DOI: 10.1111/j.1475-3995.2000.tb00201.x.

[4] Pasquale Avella, Maurizio Boccia, and Antonio Sforza. "Resource constrained shortest path problems in path planning for fleet management". In: *Journal of Mathematical Modelling and Algorithms* 3.1 (2004), pp. 1–17. ISSN: 1570-1166. DOI: 10.1023/B:JMMA.0000026675.50719.ce.

[5] Roberto Baldacci, Maria Battarra, and Daniele Vigo. "Routing a heterogeneous fleet of vehicles". In: *Operations Research/ Computer Science Interfaces Series*. Vol. 43. Springer, 2008, pp. 3–27. ISBN: 9780874216561. DOI: 10.1007/978-0-387-77778-8_1. arXiv: arXiv:1011.1669v3.

[6] Brian Coltin and Manuela Veloso. "Online pickup and delivery planning with transfers for mobile robots". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014, pp. 5786–5791. ISBN: 978-1-4799-3685-4. DOI: 10.1109/ICRA.2014.6907709.

[7] Brian Coltin and Manuela Veloso. "Ridesharing with passenger transfers". In: *Intelligent Robots and Systems (IROS 2014)*. 2014.

[8] Brian Coltin and Manuela Veloso. "Scheduling for Transfers in Pickup and Delivery Problems with Very Large Neighborhood Search". In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. Québec City, Québec, Canada, 2014, pp. 2250–2256. ISBN: 9781577356790.

[9] Rina Dechter. "Temporal Constraint Networks". In: *Constraint Processing*. Ed. by Rina Dechter. The Morgan Kaufmann Series in Artificial Intelligence. San Francisco: Morgan Kaufmann, 2003. Chap. 12, pp. 333–362. ISBN: 978-1-55860-890-0. DOI: 10.1016/B978-155860890-0/50013-X.

[10] Alexander Dettmann et al. "Heterogeneous Modules with a Homogeneous Electromechanical Interface in Multi-Module Systems for Space Exploration". In: *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA '11)*. Shanghai, China: ESA, May 2011, pp. 1964–1969.

[11] Virginia Dignum, ed. *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. IGI Global, 2009. ISBN: 9781605662572.

[12] Rodolfo Dondo and Jaime Cerdá. "A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows". In: *European Journal of Operational Research* 176.3 (2007), pp. 1478–1507. DOI: 10.1016/j.ejor.2004.07.077.

[13] Michael Drexl. "Applications of the vehicle routing problem with trailers and transshipments". In: *European Journal of Operational Research* 227.2 (2013), pp. 275–283. DOI: 10.1016/j.ejor.2012.12.015.

[14] Lester Randolph Ford and Delbert Ray Fulkerson. *Flows in networks*. Tech. rep. Santa Monica, California: The RAND Corporation, 1963, p. 152.

[15] Free Software Foundation. *GLPK (GNU Linear Programming Kit)*. Available at: https://www.gnu.org/software/glpk, (Accessed: 1 October 2015). 2015.

[16] Rina Detcher. *Constraint Processing*. Vol. 33. Morgan Kaufmann Publishers Inc., 2003, p. 480. ISBN: 978-1-55860-890-0.

[17] Thomas M. Roehr, Florian Cordes, and Frank Kirchner. "Reconfigurable Integrated Multirobot Exploration System (RIMRES): Heterogeneous Modular Reconfigurable Robots for Space Exploration". In: *Journal of Field Robotics* 31.1 (Jan. 2014), pp. 3–34. ISSN: 15564959. DOI: 10.1002/rob.21477.

[18] Thomas M. Roehr and Ronny Hartanto. "Towards safe autonomy in space exploration using reconfigurable multi-robot systems". In: *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2014)*. ESA, 2014.

[19] Thomas M. Roehr and Frank Kirchner. "Spatio-Temporal Planning for a Reconfigurable Multi-Robot System". In: *Proceedings of the 4th Workshop on Planning and Robotics (PlanRob)*. Ed. by Alberto Finzi and Erez Karpas. London, 2016, pp. 135–146.

[20] Christian Schulte and Guido Tack. "View-based propagator derivation". In: *Constraints* 18.1 (2012), pp. 75–107. ISSN: 1383-7133. DOI: 10.1007/s10601-012-9133-z. arXiv: arXiv:0908.2050v1.

[21] Roland Sonsalla et al. "Towards a Heterogeneous Modular Robotic Team in a Logistic Chain for Extraterrestrial Exploration". In: *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space*. Montreal, Canada: ESA, 2014.

[22] Paolo Toth and Daniele Vigo, eds. *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. 2nd ed. MOS-SIAM, 2014. ISBN: 1611973597. DOI: 10.1137/1.9781611973594.