A Formal Graphical Language of Interdependence in Teamwork

Wei, Changyun; Hendriks, Koen V.; van Riemsdijk, M. Birna; Jonker, Catholijn

DOI
10.1109/MIS.2019.2942580

Publication date
2019

Document Version
Final published version

Published in
IEEE Intelligent Systems

A Formal Graphical Language of Interdependence in Teamwork

Changyun Wei
Hohai University

M. Birna van Riemsdijk
Delft University of Technology

Koen V. Hindriks
Vrije Universiteit Amsterdam

Catholijn M. Jonker
Delft University of Technology

***Abstract*—Agents in teamwork may be highly interdependent on each other, and the awareness of interdependences is an important requirement for designing and consequently implementing a multiagent system. In this article, we propose a formal graphical and domain-independent language that can facilitate the identification of comprehensive interdependences among the agents in teamwork. Moreover, a formal semantics is also introduced to precisely express and explain the properties of a graphical structure. The novel feature of the graphical language is that it complements the Interdependence Analysis Color Scheme in a way that explicitly models negative influences and, in addition, provides a visual-communication aid for developers. To demonstrate the applicability and sufficiency of the graphical language in a variety of domains, our case studies include a multirobot scenario and a human-robot scenario.**

Over the past several decades, researchers have put in a great deal of effort into developing various kinds of agents from single to multiple, and significant achievements have been made to propose and validate many ad hoc or empirical mechanisms for designing a multiagent system.[1] In this article, we focus on the early design phase of a multiagent system, with the aim of producing a domain-independent formal language as a

Digital Object Identifier 10.1109/MIS.2019.2942580
Date of publication 19 September 2019; date of current version 18 November 2019.

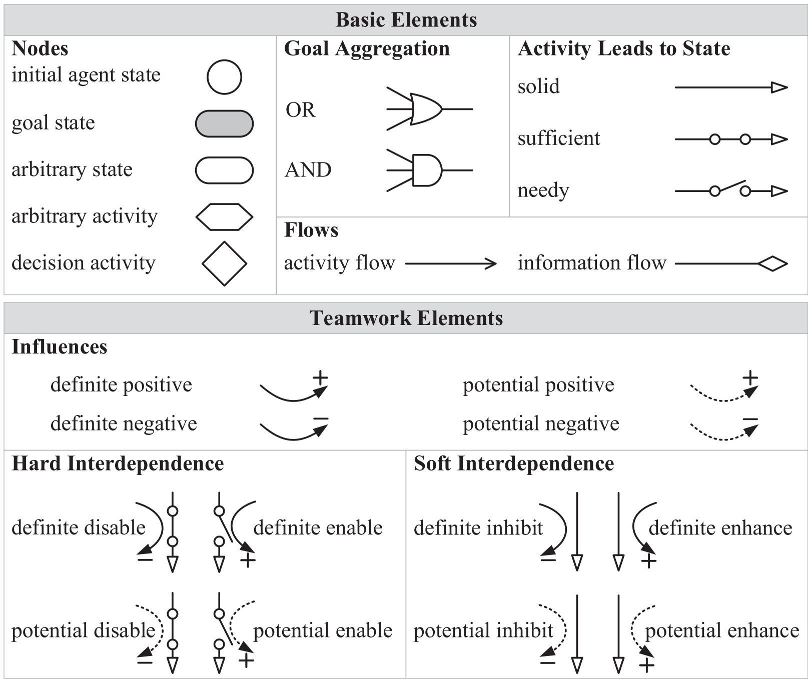


Figure 1. Overview of the formal graphical specification of elements.

modeling methodology for analyzing and consequently facilitating the implementation of a cooperative agent team.

The awareness of interdependences between all the agents engaged in teamwork is an important requirement for designing such a system because the activities performed by one agent may influence the activities of the others.[2] In many applications, some tasks are beyond the capabilities of an individual agent working alone, whereas the others can be done by a single agent but more members can improve team performance. The use of multiple cooperative agents may yield performance gains, but in order to realize such gains the agents need to systematically coordinate with each other. Otherwise, team performance can be impaired due to potential interference among members. Thus, the awareness of interdependencies is helpful and must be taken into consideration in a design of a cooperative agent team.

The main contribution of our work is a *formal graphical and domain-independent language* for modeling comprehensive interdependences in teamwork. Our work complements the Interdependence Analysis Color Scheme (IACS)[3] in a way that explicitly models negative influences among members and, moreover, introduces a formal semantics to precisely express and explain the properties of a graphical structure. As a graphical tool, it can be easily used as a visual-communication aid such as UML diagrams, Petri Nets,[4] and other popular agent-oriented methodologies (such as Tropos[5] and Prometheus,[6]) and in particular our graphical language can facilitate the identification of both positive (supporting) and negative (interfering) interdependence relationships in the early design phase of a multiagent system for developers.

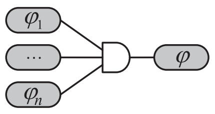
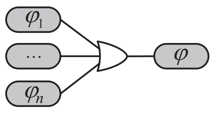
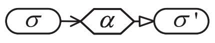
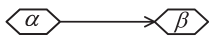
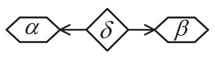
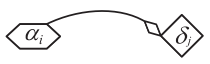
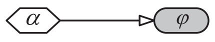
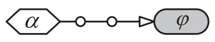
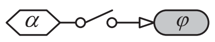
LANGUAGE SPECIFICATION

The overview of the elements of the proposed language is shown in Figure 1. The upper part of the figure lists all the basic elements, i.e., nodes and edges, whereas the lower part summarizes the elements for modeling influences and interdependences in teamwork. A resulting diagram, depicted by the proposed language, is a graph with various types of nodes and edges. The key concepts of the graphical language are described as follows.

Agent Concept

Agents, as usual, perform activities to change environments or respond to teammates. We use

Table 1. Graphical diagrams and semantics for goal aggregation, composed activities, and communication.

	Graphical Diagrams	Expressions	Meanings
Goal Diagrams		AndG(φ,Γ)	In order to achieve goal φ, all the input subgoals $\Gamma=\{\varphi_1,\cdots,\varphi_n\}$ must be achieved.
		OrG(φ,Γ)	In order to achieve goal φ, at least one of the subgoals $\Gamma=\{\varphi_1,\cdots,\varphi_n\}$ must be achieved.
State Transition		$\sigma\xrightarrow{\alpha}\sigma'$	Performing activity α at state σ leads to state σ'.
Activity Diagrams		$(\alpha;\beta)$	Activity α is followed by activity β.
		$\delta(\alpha\mid\beta)$	δ is performed to decide whether to perform activity α or β.
		(comm(α_i) ;δ_j)	Communication performed as part of activity α by agent i will affect decision δ of agent j.
Activity-Goal Diagrmas		SoliAc(α,φ)	α is a solid activity to achieve goal φ.
		SuffAc(α,φ)	α is a sufficient activity to achieve goal φ.
		NeedAc(α,φ)	α is a needy activity to achieve goal φ.

a *circle* node to represent the initial state of an agent. If a diagram has a node of an agent's initial state, an *activity flow* must come out of it, pointing to an activity node. We use $\mathcal{A}$ to represent a finite set of agents, and i as a variable denotes some arbitrary agent. In a resulting diagram, each node can be labeled so that we can distinguish them. For instance, we put i into a circle node to represent the initial state of the ith agent.

States, Properties, and Goals

In our language, a *state* means the state of the environment, and we use Σ to represent the set of all the states with typical element $\sigma\in\Sigma$ representing an arbitrary state. Properties of states are indicated by $\mathcal{P}$ with typical element $p\in\mathcal{P}$. We assume an environmental relation $\sigma\models p$, which defines when property p holds in the environment. *Goals* indicate the desired states of the environment that can be realized by agents performing corresponding activities. We use $\Phi\subseteq\mathcal{P}$ to denote the set of goals, with typical element $\varphi\in\Phi$ representing an arbitrary goal.

In our graphical language, we use a *rounded rectangle* to represent an arbitrary state, and a *shaded rounded rectangle* to denote a node of goal state. If there is no confusion over a goal state node, we call it a goal node for simplicity. A goal node in a diagram can represent the team goal or a subgoal that contributes to the team goal. Sometimes, a complex subgoal can also be divided into smaller subgoals. In Tropos,[5] abstract goals need to be refined into more concrete leaf-goals, so we also define two logic AND/OR gates for goal aggregation (see Table 1):

Definition 1 (AND goal). *If the goal φ is connected to a set Γ of input goals by a logic AND gate, the fulfillment of φ requires that all the input goals must be satisfied, and we say that φ is an AND goal*

$$\mathbf{AndG}(\varphi,\Gamma) \overset{\text{def}}{=} (\forall\varphi' \in \Gamma : \sigma \models \varphi') \Rightarrow \sigma \models \varphi.$$

Definition 2 (OR goal). *If the goal φ is connected to a set Γ of input goals by a logic OR gate, the fulfillment of φ requires that at least one of input goals must be satisfied, and we say φ is an OR goal*

$$\mathbf{OrG}(\varphi,\Gamma) \overset{\text{def}}{=} (\exists\varphi' \in \Gamma : \sigma \models \varphi') \Rightarrow \sigma \models \varphi.$$

Activity Concept

Activities are performed by agents to change the environments. A *hexagon* node represents an arbitrary activity, and we use $\mathcal{A}c$ to denote the set of activities that the agents can perform, with typical elements of α or β. In order to distinguish which agent performs what activity, we use, for example, α_i to express that agent i performs activity α. If there is no confusion over an activity, we drop the agent's index for simplicity throughout this article.

State Transition. The transitional model of performing activities is abstracted from the external environment, and structured in the form of $\sigma \xrightarrow{\alpha} \sigma'$ expressing that performing activity α at state σ leads to state σ' (see Table 1). We define *reliability* and *efficiency* to describe the fulfillment of a goal and the efficiency of achieving a goal, respectively.

Definition 3 (Reliability). *If activity α has the* reliability *to achieve the goal φ at state σ, then $\sigma \xrightarrow{\alpha} \sigma' \Rightarrow \sigma' \models \varphi$.*

Definition 4 (Efficiency). *If activity α is more efficient than activity β in achieving goal φ, then* $\mathbf{Effic}(\sigma \xrightarrow{\alpha} \sigma') > \mathbf{Effic}(\sigma \xrightarrow{\beta} \sigma'')$*, such that $\sigma' \models \varphi$, $\sigma'' \models \varphi$,* $\mathbf{Effic}(\sigma \xrightarrow{\alpha} \sigma') \in \mathbb{R}$ *and* $\mathbf{Effic}(\sigma \xrightarrow{\beta} \sigma'') \in \mathbb{R}$.

Activity Composition. The activities that the agents can perform include action execution, decision making, and communication. If $\alpha, \beta \in \mathcal{A}c$ are activities, the following expressions are also activities:

- $(\alpha;\beta) \in \mathcal{A}c$ expresses that activity α is followed by activity β,
- $(\alpha||\beta) \in \mathcal{A}c$ expresses that activity α is performed in parallel with activity β, and
- $(\alpha|\beta) \in \mathcal{A}c$ expresses that either activity α or β will be performed.

As in Table 1, arbitrary state nodes can be put before and after an activity node to highlight state transitions. For simplicity, we can drop them. In this article, if performing an activity influences the execution of the other activity, we assume that those activities are performed in parallel. Thus, we use $\sigma \xrightarrow{\alpha||\beta} \sigma'$ to express that performing activity α and β concurrently at state σ leads to state σ'. In Table 1, we do not depict the composed parallel activities, i.e., $(\alpha||\beta)$, which will be discussed for teamwork.

Decision activity. We use a *diamond* (or rhombus) to represent a decision activity that indicates a test condition in which the agent needs to evaluate its current situation to make choices on what to do next. One way of using a decision activity node is the same as in flow charts, where a decision is necessary, and usually two arrows of activity flow coming out of it (i.e., Yes/No questions or True/False tests). More than two arrows of activity flow are also allowed. If there is a decision activity node in a diagram, the corresponding agent needs to make a decision.

Communication activity. Performing activity α by agent i will affect decision δ of agent j via communication is expressed by $(\mathbf{comm}(\alpha_i);\delta_j)$. If there is a communication activity in a diagram, an arrow of information flow comes out of it, pointing to a decision activity node of another agent. A connecting edge of information flow goes to a decision activity node because such a structure can directly express what decisions will be affected. Mutual communication is also allowed between two decision activity nodes.

Activity Leads to Achieving a Goal. When an agent performs an activity for the achievement of a goal, the reliability or efficiency of the activity may be influenced by other activities. The influences can be *positive* or *negative*. A positive influence improves the reliability or efficiency of the activity, while a negative one impairs its reliability or efficiency. In order to model stochastic effects, the influences can be *definite* or *potential*.

Definition 5 (Solid activity). *Activity α is considered as a* solid activity *for the fulfillment of goal φ at a state, if any other activities cannot impair its* reliability

$$\mathbf{SoliAc}(\alpha,\varphi) \overset{\text{def}}{=} \forall\beta \in \mathcal{A}c, \forall\sigma,\sigma' \in \Sigma : \\ \sigma \xrightarrow{\beta||\alpha} \sigma' \Rightarrow \sigma' \models \varphi.$$

A solid activity can always achieve its intended goal regardless of what happens, but its *efficiency* can be influenced by other activities.

Definition 6 (Sufficient activity). *Activity α is considered as a* sufficient activity *for the fulfillment of goal φ at a state, if it is sufficient to achieve the goal, but its* reliability *can be impaired by other activities*

$$\mathbf{SuffAc}(\alpha,\varphi) \overset{\text{def}}{=} \forall\sigma,\sigma' \in \Sigma : \\ \sigma \overset{\alpha}{\rightarrow} \sigma' \Rightarrow \sigma' \models \varphi \text{ and} \\ \exists\beta \in \mathcal{A}c, \exists\sigma'' \in \Sigma : \sigma \xrightarrow{\beta||\alpha} \sigma'' \Rightarrow \sigma'' \not\models \varphi.$$

A sufficient activity can be interrupted by other activities and become impossible for achieving the goal. Thus, when we draw a sufficient activity, there is a *closed switch* along the arrow linking the activity to its intended goal. It means that there are some other activities that can *definitely* or *potentially* open the switch, implying that performing the activity may not achieve the goal any more due to such negative influences.

Definition 7 (Needy activity). *An activity α is considered as a* needy activity *for the fulfillment of goal φ at a state, if it cannot achieve the goal by performing the activity alone*

$$\mathbf{NeedAc}(\alpha,\varphi) \overset{\text{def}}{=} \exists\beta \in \mathcal{A}c, \forall\sigma,\sigma',\sigma'' \in \Sigma : \\ \sigma \overset{\alpha}{\rightarrow} \sigma' \Rightarrow \sigma' \not\models \varphi \text{ and } \sigma \xrightarrow{\beta||\alpha} \sigma'' \Rightarrow \sigma'' \models \varphi.$$

A needy activity is insufficient to achieve a goal alone, but it can become possible with the help of other activities. Thus, an *open switch* is placed along the arrow linking the activity to its intended goal, which means that the activity cannot directly lead to the fulfillment of the goal, but the switch can be closed with the help of other activities.

INTERDEPENDENCE MODELING

Interdependencies are inherent in teamwork, where one agent may affect the ability of the other to achieve a goal or execute a plan.[7] A simple interpretation of interdependence is given by Malone and Crowston,[8] where interdependence means mutual dependence, and coordination for multiagent teamwork is considered as the process of dealing with dependencies among the activities of the agents in a team. Interdependencies are analyzed in terms of roles.[9] A more comprehensive definition of interdependence, called IACS,[3] is presented to model how one agent *support* the other in teamwork. However, it is still hard to identify what needs to be done for each agent in a resulting color table.

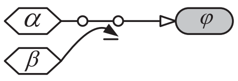
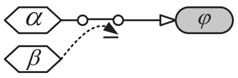
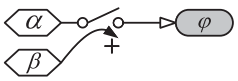
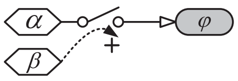
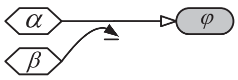
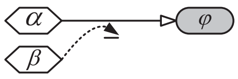
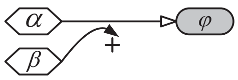
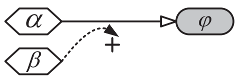
Due to the presence of multiple agents in a team, one agent's activity may also *hamper* the reliability or efficiency of another agent's activity. In order for designers to identify potential *interference* in teamwork, our graphical language, additionally, intends to *explicitly model the negative interdependences* (see Table 2). In IACS, interdependences are categorized into "hard" (absolutely necessary for carrying out the joint activity) or "soft" (defining possible opportunities for improving joint activity), but in our work hard and soft interdependences are characterized by the *reliability* and *efficiency* of achieving a goal, respectively.

Hard Interdependence

Hard interdependences are associated with a sufficient or a needy activity, concerning the *reliability* of achieving a goal. To be more specific, performing a *sufficient activity* can achieve a goal alone, but the fulfillment can be interrupted by other activities. In contrast, performing a *needy activity* cannot achieve a goal alone, but it can become possible with the help of other activities. In a diagram, hard interdependences always exert positive or negative influences on a *switch* that is associated with a sufficient or a needy activity. In this article, we categorize four types of hard interdependences.

Definition 8 (Definitely disable). *If performing β exerts a* definitely negative *effect on the* reliability *of performing a sufficient activity α in achieving goal φ at a state, we say that activity β*

Table 2. Graphical diagrams and expressions for interdependences.

	Graphical Diagrams	Expressions	Meanings
Hard Interdependence		DefDis(β,α,φ)	β will definitely disable α for the fulfilment of goal φ.
		PotDis(β,α,φ)	β will potentially disable α for the fulfilment of goal φ.
		DefEna(β,α,φ)	β will definitely enable α for the fulfilment of goal φ.
		PotEna(β,α,φ)	β will potentially enable α for the fulfilment of goal φ.
Soft Interdependence		DefInh(β,α,φ)	β will definitely inhibit α for the fulfilment of goal φ.
		PotInh(β,α,φ)	β will potentially inhibit α for the fulfilment of goal φ.
		DefEnh(β,α,φ)	β will definitely enhance α for the fulfilment of goal φ.
		PotEnh(β,α,φ)	β will potentially enhance α for the fulfilment of goal φ.

will definitely disable *the sufficient activity α for the fulfillment of the goal φ*

$$\mathbf{DefDis}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \mathbf{SuffAc}(\alpha,\varphi) \text{ and}$$
$$\forall \sigma, \sigma' \in \Sigma : \sigma \xrightarrow{\beta\|\alpha} \sigma' \Rightarrow \sigma' \not\models \varphi.$$

For example, if two same sized robots push a box in opposite directions, they will get stuck because one robot's action may definitely disable the action execution of the other.

Definition 9 (Potentially disable). *If performing β exerts a* potentially negative *effect on the* reliability *of performing a sufficient activity α in achieving goal φ at a state, we say that activity β will* potentially disable *the sufficient activity α for the fulfillment of the goal φ*

$$\mathbf{PotDis}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \mathbf{SuffAc}(\alpha,\varphi) \text{ and}$$
$$\forall \sigma, \exists \sigma' \in \Sigma : \sigma \xrightarrow{\beta\|\alpha} \sigma' \Rightarrow \sigma' \not\models \varphi.$$

For instance, if two unmanned cars are driving on a highway in parallel at a very close distance, one car may potentially disable the driving of the other.

Definition 10 (Definitely enable). *If performing β exerts a* definitely positive *effect on the* reliability *of performing a needy activity α in achieving goal φ at a state, we say that activity β will* definitely enable *the needy activity α for the fulfillment of the goal φ*

$$\mathbf{DefEna}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \mathbf{NeedAc}(\alpha,\varphi) \text{ and}$$
$$\forall \sigma, \sigma' \in \Sigma : \sigma \xrightarrow{\beta\|\alpha} \sigma' \Rightarrow \sigma' \models \varphi.$$

For example, a robot that cannot push a heavy box alone, can do so with the help of a teammate.

Definition 11 (Potentially enable). *If performing β exerts a* potentially positive *effect on the* reliability *of performing a needy activity α in achieving goal φ at a state, we say that activity β will* potentially enable *the needy activity α for the fulfillment of the goal φ*

$$\mathbf{PotEna}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \mathbf{NeedAc}(\alpha,\varphi) \text{ and}$$
$$\forall \sigma, \exists \sigma' \in \Sigma : \sigma \xrightarrow{\beta\|\alpha} \sigma' \Rightarrow \sigma' \models \varphi.$$

For example, robot A does not have the ability to open a door so as to go through it, but can achieve this by closely following robot B that performs the open activity.

Soft Interdependence

Soft interdependences are associated with a solid or a sufficient activity, concerning the *efficiency* of achieving a goal. As mentioned above, a solid activity is always robust to achieve its intended goal, but its efficiency can be influenced. In Table 2, we do not list the cases for sufficient activities because soft interdependences always exert positive or negative influences on the connecting line behind an arrow (not on a switch). In this article, we categorize four types of soft interdependences.

Definition 12 (Definitely inhibit). *If performing β exerts a* definitely negative *effect on the* efficiency *of performing activity α in achieving goal φ at a state, we say that activity β will* definitely inhibit *activity α for the fulfillment of goal φ*

$$\mathbf{DefInh}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \forall\sigma,\sigma',\sigma'' \in \Sigma :$$
$$\mathbf{Effic}(\sigma \xrightarrow{\beta||\alpha} \sigma') < \mathbf{Effic}(\sigma \overset{\alpha}{\rightarrow} \sigma'').$$

For example, although a small robot cannot stop a large robot from pushing a box, it can slow down the movement and increase the execution time.

Definition 13 (Potentially inhibit). *If performing β exerts a* potentially negative *effect on the* efficiency *of performing activity α in achieving goal φ at a state, we say that activity β will* potentially inhibit *the activity α for the fulfillment of the goal φ*

$$\mathbf{PotInh}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \forall\sigma,\exists\sigma',\sigma'' \in \Sigma :$$
$$\mathbf{Effic}(\sigma \xrightarrow{\beta||\alpha} \sigma') < \mathbf{Effic}(\sigma \overset{\alpha}{\rightarrow} \sigma'').$$

For example, when robot A navigates towards its destination, the existence of robot B may potentially increase the path cost of robot A.

Definition 14 (Definitely enhance). *If performing β exerts a* definitely positive *effect on the* efficiency *of performing activity α in achieving goal φ at a state, we say that activity β will* definitely enhance *the activity α for the fulfillment of the goal φ*

$$\mathbf{DefEnh}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \forall\sigma,\sigma',\sigma'' \in \Sigma :$$
$$\mathbf{Effic}(\sigma \xrightarrow{\beta||\alpha} \sigma') > \mathbf{Effic}(\sigma \overset{\alpha}{\rightarrow} \sigma'').$$

For example, one robot can help another to speed up the task execution.

Definition 15 (Potentially enhance). *If performing β exerts a* potentially positive *effect on the* efficiency *of performing activity α in achieving goal φ at a state, we say that activity β will* potentially enhance *the activity α for the fulfillment of the goal φ*

$$\mathbf{PotEnh}(\beta,\alpha,\varphi) \overset{\text{def}}{=} \forall\sigma,\exists\sigma',\sigma'' \in \Sigma :$$
$$\mathbf{Effic}(\sigma \xrightarrow{\beta||\alpha} \sigma') > \mathbf{Effic}(\sigma \overset{\alpha}{\rightarrow} \sigma'').$$

For example, one robot broadcasts its current locations and intended locations may potentially help another robot to optimize collision-free paths in congested situations.

To analyze a design of a multiagent system, we can depict a diagram using the proposed graphical language. Hard and soft interdependencies can facilitate the identification of positive (supporting) and negative (interfering) relationships, and a resulting diagram can point out what kinds of interdependences happen to what elements. Afterwards, when implementing the system, the developers can pay more attention to those elements so as to improve the resilience (i.e., the stability and adaptability[3,10]) of the system.

CASE STUDIES

Multirobot Foraging

Multirobot foraging is a canonical task that can be motivated by many practical applications such as search and rescue, hazardous waste clean-up, and automated warehouse systems. Here, we use a simple scenario for illustration, where a red, a green, and a blue box are dispersed in an unknown environment, and two robots, named 1 and 2, have to explore the environment so as to find and deliver them back to a home base. To design such a multiagent system, developers can produce a goal-oriented or an action-oriented solution, according to their preferences.

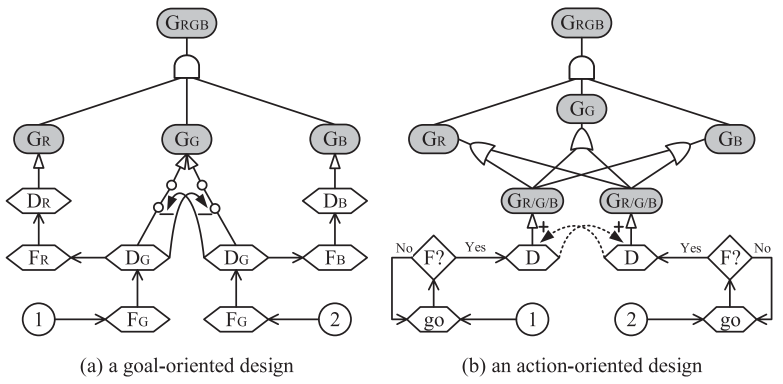


Figure 2. Goal-oriented design versus action-oriented design for multirobot foraging.

Figure 2(a) demonstrates a goal-oriented design based on teamwork models, depicting the result of a task allocation mechanism that generates plan-based actions. In order to achieve the team goal G_{RGB}, the depicted solution instructs robot 1 and 2 to first perform F_G and D_G (i.e., find and deliver the green box) at the same time. Developers can easily notice that the two paths are only sufficient to achieve the subgoal G_G because the robots can definitely disable each other's subgoal (as the environment only has one green box). As a result, they cannot continue with the subsequent subgoals (i.e., G_R or G_B). In addition, developers can get an insight that the task allocation mechanism should avoid allocating a subgoal to more than one robot in such a scenario.

Figure 2(b) shows an action-oriented design, in which each robot performs reactive actions to search and find boxes without having any specific intended goal. Specifically, if a robot finds a box at a place, then it will deliver it to achieve a subgoal. Developers can get a general impression that both of the robots are solid to complete the team goal alone, but two robots may potentially enhance team performance (i.e., reduce the completion time) by performing the task in a fully decentralized manner.

DARPA Robotics Challenge (DRC)

The DRC is a robotic competition, aiming at developing ground robots that can assist humans to perform complex tasks in dangerous human-engineered environments. The DRC is divided into several tracks, and we take a slice of the virtual robotics challenge (VRC) track for demonstration. In the task, an operator needs to *remotely control a simulated humanoid robot* to find the hose and pick it up. We take this task as an example for case studies because the IACS[3] also uses this task for evaluation, so we can make a direct comparison.

Figure 3 shows how the IACS analyzes the task by a color table. The colors of the "performer" columns describe the capacity of the performer, while the colors of the "supporting team members" columns are an assessment of team member's potential to support the performer. In the color table, green means the robot/human can do the task; yellow means the robot/human can do it but less than perfect reliability; orange means the robot/human can contribute but need assistance; red means the robot/human cannot do it. Although such a color pattern can express the required *capacities* for each subtask, it is still hard to specify the respective activities that the robot/human has to perform in teamwork. Moreover, the color table cannot concisely describe how one supports the other, e.g., the four columns are all yellow in the row "position the hand for grasping." To do this, the IACS needs supplementary paragraphs for further explanation in the article by Johnson *et al.*[3]

Comparatively, as shown in Figure 4, our graphical language can straight-forwardly depict what the human/robot has to do, which functions are automated for the robot to perform by itself, as well as how the human remotely controls the robot in teamwork. For example, in order to achieve the subgoal "locate the hose," the robot can sense the hose but need the human's assistance in recognizing the hose. Thus, the resulting diagram indicates that hose recognition algorithm should be developed for the robot in the future. We can also find that the

Tasks	Hierarchical Sub-tasks	Required Capacities	Team Member Role Alternatives			
			Alternative 1		Alternative 2	
			Performer	Supporting Team Members	Performer	Supporting Team Members
			Robot	Human	Human	Robot
Find the hose and pick it up	Locate the hose	Sense the hose	🟩	🟥	🟩	🟥
		Recognize the hose	🟥	🟥	🟩	🟩
	Walk up to the hose	Understand the arm workspace	🟩	🟥	🟨	🟨
		Know where to stand for picking up the hose	🟧	🟧	🟨	🟨
		Specify the footsteps to position the robot	🟨	🟨	🟨	🟨
		Execute the footsteps	🟩	🟥	🟥	🟥
	Grasp the hose	Know how to grasp the hose	🟥	🟥	🟨	🟨
		Position the hand for grasping	🟨	🟨	🟨	🟨
		Execute the grasp	🟩	🟥	🟩	🟩
		Verify the grasp is good	🟥	🟥	🟨	🟨
	Lift the hose	Raise the hose for carrying	🟩	🟩	🟩	🟩
		Abort if grasp is improper	🟥	🟥	🟩	🟩

Figure 3. Interdependence analysis for VRC subtask using the color scheme.[3]

robot is sufficient to position the hand for grasping (occasionally it fails to achieve this), but the operator's activity of remotely control the hand joints can potential enhance efficiency (here "potential" indicates that occasionally the robot can do better by itself). By investigating the resulting diagram, the developers can easily notice the elements related to hard or soft interdependences, and then identify which functions still need to be developed and refined so as to realize higher levels of autonomy in future.

CONCLUSION

In this article, we introduced a formal graphical and domain-independent language for modeling comprehensive interdependences of a multiagent system in the early design phase. As a modeling tool, it allows developers to analyze various design choices to improve development efficiency in building highly interdependent systems. The benefits of higher levels of autonomy cannot be realized without addressing interdependence through cooperation and collaboration among agents. For example, before a fully autonomous car is developed, several levels of the automated driving system have to be designed to assist the driver. At each level, the developers should identify who is in control/assistance in what situations or who is assigned to what tasks, so they can deftly make appropriate tradeoffs in allocating tasks to either the driver or the automated driving system.

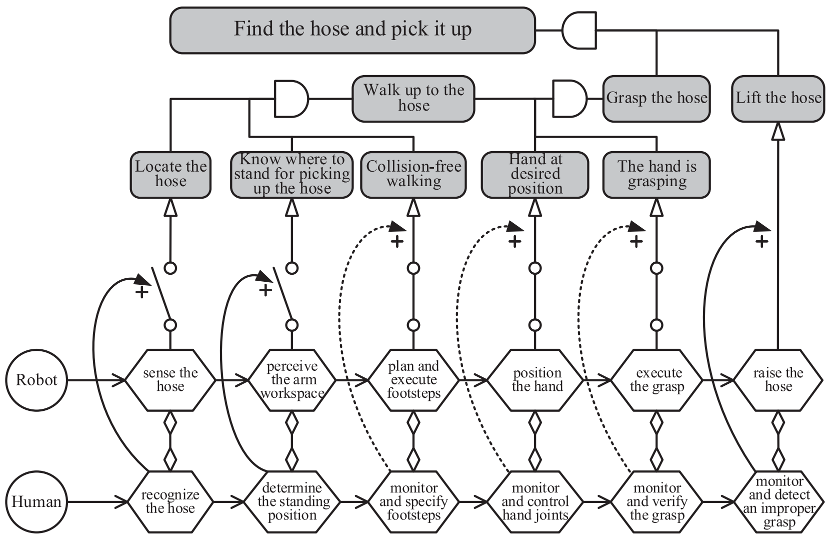

Find the hose and pick it up
Walk up to the hose
Grasp the hose
Lift the hose
Locate the hose
Know where to stand for picking up the hose
Collision-free walking
Hand at desired position
The hand is grasping
+
+
+
+
+
+
Robot
sense the hose
perceive the arm workspace
plan and execute footsteps
position the hand
execute the grasp
raise the hose
Human
recognize the hose
determine the standing position
monitor and specify footsteps
monitor and control hand joints
monitor and verify the grasp
monitor and detect an improper grasp

Figure 4. Interdependence analysis for VRC subtask using the graphical language.

In future work, we will apply our proposed graphical modeling tool to design sophisticated multirobot human systems, e.g., unmanned surface/ground vehicles, in which various levels of unmanned systems have to work together with their operators. The use of our graphical modeling tool can facilitate the uncovering of the system requirements and detailed specifications. Moreover, we also would like to prove properties of example designs using formalism and investigate the scalability and composition of a design.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61703138 and in part by the Natural Science Foundation of Jiangsu Province under Grant BK20170307.

REFERENCES

1. G. Hollinger, S. Singh, J. Djugash, and A. Kehagias, “Efficient multi-robot search for a moving target,” *Int. J. Robot. Res.*, vol. 28, no. 2, pp. 201–219, 2009.
2. M. Johnson, J. M. Bradshaw, P. Feltovich, C. Jonker, B. van Riemsdijk, and M. Sierhuis, “Autonomy and interdependence in human-agent-robot teams,” *IEEE Intell. Syst.*, vol. 27, no. 2, pp. 43–51, Mar./Apr. 2012.
3. M. Johnson, J. M. Bradshaw, P. J. Feltovich, C. M. Jonker, M. B. Van Riemsdijk, and M. Sierhuis, “Coactive design: Designing support for interdependence in joint activity,” *J. Human-Robot Interact.*, vol. 3, no. 1, pp. 43–69, 2014.
4. S. Pujari and S. Mukhopadhyay, “Petri net: A tool for modeling and analyze multi-agent oriented systems,” *Int. J. Intell. Syst. Appl.*, vol. 4, no. 10, pp. 103–112, 2012.
5. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, “Tropos: An agent-oriented software development methodology,” *Auton. Agents Multi-Agent Syst.*, vol. 8, no. 3, pp. 203–236, 2004.
6. L. Padgham and M. Winikoff, “Prometheus: A practical agent-oriented methodology,” in *Agent-Oriented Methodologies*, Hershey, PA, USA: IGI Global, 2005, pp. 107–135.
7. N. R. Jennings, “Coordination techniques for distributed artificial intelligence,” *Found. Distrib. Artif. Intell.*, vol. 187, pp. 187–210, 1996.
8. T. Malone and K. Crowston, “The interdisciplinary study of coordination,” *ACM Comput. Surv.*, vol. 26, no. 1, pp. 87–119, 1994.
9. E. Steegmans, D. Weyns, T. Holvoet, and Y. Berbers, “A design process for adaptive behavior of situated agents,” in *Agent-Oriented Software Engineering V*, Berlin, Germany: Springer, 2005, pp. 109–125.
10. E. K. Chiou and J. D. Lee, “Cooperation in human-agent systems to support resilience: A microworld experiment,” *Human Factors*, vol. 58, no. 6, pp. 846–863, 2016.

Changyun Wei is currently an Associate Professor with the College of Mechanical and Electrical Engineering, Hohai University, Nanjing, China. His research interests include cognitive robotics, multi-agent/robot systems. He received the Ph.D. degree with the Interactive Intelligence Group, Department of Intelligent Systems, Delft University of Technology, Delft, The Netherlands. He is a member of the International Society of Applied Intelligence, Chinese Association for Artificial Intelligence. He is the corresponding author and can be contacted at: weichangyun@hotmail.com.

Koen V. Hindriks is currently a Full Professor with the Artificial Intelligence Group, Vrije Universiteit Amsterdam, The Netherlands. His research focuses on the analysis, modeling, and development of agent technology. He received the Ph.D. degree in intelligent systems from Utrecht University, Utrecht, The Netherlands. Contact him at: k.v.hindriks@vu.nl.

M. Birna van Riemsdijk is currently an Associate Professor with the Interactive Intelligence Group, Delft University of Technology, Delft, The Netherlands. She is interested in technology that plays an active role in human connection. She received a Ph.D. degree in intelligent systems from Utrecht University, Utrecht, The Netherlands. Contact her at: m.b.vanriemsdijk@tudelft.nl.

Catholijn M. Jonker is currently a Full Professor with the Interactive Intelligence Group, Delft University of Technology, Delft, The Netherlands. Her research interests include negotiation, teamwork, and the dynamics of individual agents and organizations. She received a Ph.D. degree from Utrecht University, Utrecht, The Netherlands. She is the fellow and board member of the European Association for Artificial Intelligence. Contact her at: c.m.jonker@tudelft.nl.