Auton Agent Multi-Agent Syst
DOI 10.1007/s10458-008-9032-2

Dynamic intention structures I: a theory of intention representation

Luke Hunsberger · Charles L. Ortiz, Jr.

Abstract This article introduces a new theory of intention representation which is based on a structure called a Dynamic Intention Structure (DIS). The theory of DISs was motivated by the problem of how to properly represent incompletely specified intentions and their evolution. Since the plans and intentions of collaborating agents are most often elaborated incrementally and jointly, elaboration processes naturally involve agreements among agents on the identity of appropriate agents, objects and properties that figure into their joint plans. The paper builds on ideas from dynamic logic to present a solution to the representation and evolution of agent intentions involving reference to incompletely specified and, possibly, mutually dependent intentions, as well as the objects referenced within those intentions. It provides a first order semantics for the resulting logic. A companion paper extends further the logical form of DISs and explores the problem of logical consequence and intention revision.

Keywords Intentions · Representation · Collaborative planning

1 Introduction

Numerous theories of intention and collaboration have been developed over the past 25 years [4,5,7,14,15,21,22,30,32,33,36]. However, the representation of the content of agent intentions has not addressed the following important issues. First, agents frequently need to communicate with others about parameters in their plans (e.g., when negotiating over constraints on parameters). Thus, the representation of such parameters must enable agents to unambiguously refer to them when communicating with other agents, even if those parameters have not yet received values. Second, agents frequently delegate parameter-binding decisions in

L. Hunsberger (✉)
Vassar College, Poughkeepsie, NY, USA
e-mail: hunsberg@cs.vassar.edu

C. L. Ortiz
SRI International, Menlo Park, CA, USA
e-mail: ortiz@ai.sri.com

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

collaborative activity. Thus, the representation of parameters in the content of an agent's intention must be able to distinguish cases where the agent is free to select the value of a parameter from cases where the agent's intention involves some parameter whose value shall be selected by some other agent. Third, a group of collaborating agents frequently delegate subsidiary tasks or goals to one or more group members. Thus, the representation of the content of intentions must be able to properly reflect the hand-off of responsibility for the subsidiary tasks or goals while still maintaining the higher-level intention "that the selected agent do the subsidiary task or accomplish the subsidiary goal." Fourth, the representation of the content of agent intentions needs to be able to address the partiality of agent intentions and plans. For example, an agent might intend to rent a car without having a particular car in mind. Fifth, the representation of the content of agent intentions needs to accommodate the evolution of agent intentions and plans over time. Thus, a basic suite of intention-update operators must be provided. Sixth, intentions in any kind of complex task are frequently related to one another. For example, if I intend to get a car by renting it, but then find out that renting a car is impossible, then I would typically revert to my original intention to get a car, which would lead me to find other ways of getting a car (e.g., by borrowing one). Thus, the representation of intentions must address the relationships between intentions in a complex plan.

This article presents a theory of intention representation that provides solutions to these representational problems. Insodoing, it fills an important gap in existing theories of agents, planning and collaborative planning. In a companion paper [27] we present a theory of intention revision based on this representation.

Note: Throughout this article, we use the term *intention update* to refer to the incremental modification of a single intention, as opposed to *intention revision* which, in the companion paper, refers to the process of modifying the contents of an agent's entire database of intentions to accommodate the updating of a single intention or the insertion of a new intention.

2 Overview of our approach

Suppose Alice and Bob plan to travel to California together by car. Suppose further that, as part of their plan, Alice intends to rent a car and Bob intends to drive *that car* (with both of them in it) to California. How can the content of their intentions be formally represented? A straightforward approach to representing Alice's intention might employ a formula such as

$$\mathit{IntTh}(\mathtt{A}, (\exists x)\mathit{Rent}(\mathtt{A}, x) \wedge \mathit{Car}(x))$$

where *IntTh* is a modal intention operator, $\mathtt{A}$ is a constant denoting Alice, and the existentially quantified variable x represents an as-yet-unspecified car. The problem with such a representation of Alice's intention is that the scope of the existentially quantified variable is closed—and embedded within an intention operator—and, thus, unavailable for use when representing Bob's intention. However, the representation of Bob's intention must contain a reference to that same car.

Similar representational problems have been encountered when attempting to analyze sequences of sentences in a natural language. For example, consider the two sentences given below:

(1) A man walked into my room. (2) He was tall.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

A straightforward translation of the first sentence into first-order logic (FOL) might yield a formula such as: $(\exists x)Man(x) \wedge WalkedInto(x, \texttt{MyRoom})$. However, the scope of the existentially quantified variable, x, being closed complicates matters for the analysis of the second sentence. One would like to use a formula such as $Tall(x)$ for the second sentence, but x is free in that formula.

Kamp [20] developed Discourse Representation Theory (DRT) to address these kinds of quantifier scoping and reference problems in linguistics, which had been resistant to solution using first-order logic.[1] In DRT, certain data structures are used to represent the current state of a discourse (i.e., the information that can be derived from the current point in a discourse). The semantics of the data structures is given in terms of so-called *verifying embeddings;* alternatively a translation can be provided directly from discourse representation structures to formulas in first-order logic. Our approach to the representation of intentions is similar to that of DRT in that we use a certain kind of data structure—called a Dynamic Intention Structure (DIS)—to represent the content of an agent's intention. As in DRT, we define the semantics of our DIS structures by providing a translation function that maps DISs to logical formulas—in our case, a version of first-order modal logic developed by Fitting and Mendelsohn [8].

Although our approach draws from DRT, there are several significant differences. First, the content of agent intentions is structured into a variety of different fields, each of which plays a different role in the translation of that structure into logic. Second, the treatment of parameters is richer in that it clearly distinguishes parameters for which the intending agent will select values from those for which the values will be selected by some other agent or group. Third, the content distinguishes actions that the agent intends to do from propositions that the agent intends should hold. Fourth, the content includes subsidiary boxes, representing portions of a group activity that will be the responsibility of member agents or subgroups. Many of these differences are in response to the different needs to which these structures are being put in our work. Naturally, the intentions constituting a collaboration among a group of agents are different from the sentences constituting a discourse.

2.1 Overview of the syntax of dynamic intention structures

A Dynamic Intention Structure (DIS) comes in two varieties, containing the fields shown in the boxes in Fig. 1. The box on the right, called a DIS* structure, contains an extra field, *Grp*, that specifies the group in a collaborative activity. The *Agt* field specifies the agent holding the intention. The *ID* field specifies an identifier for the intention. The *ExVars* and *DefVars* fields specify two kinds of parameters: those the intending agent is free to find values for and those whose values will be determined by some other agent(s). The *ActType* field specifies the kind of action the agent intends to be done. The *Conds* field specifies conditions (or constraints) that the agent intends shall hold.[2] The *SubBoxes* field contains subsidiary DIS structures corresponding to subsidiary tasks or goals. The syntax of DISs will be given formally in Sect. 3, after presenting a series of motivating examples; however, we must first say a few words about semantics.

[1] Heim [18] presented a similar approach based on what she calls *file change semantics*. Groenendijk and Stokhof [12] presented still another approach based on Dynamic Predicate Logic.

[2] The *Conds* field contains a set of *intended conditions* (i.e., propositions that the agent intends shall hold). In contrast, this paper does not address *conditional intentions* (i.e., intentions conditioned on some proposition). Conditional intentions are treated in the companion paper [27].

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

DIS	DIS*
ID/Agt:	*ID/Agt / Grp*:
ExVars:	*ExVars*:
DefVars:	*DefVars*:
ActType:	*ActType*:
SubBoxes:	*SubBoxes*:
Conds:	*Conds*:

Fig. 1 Dynamic Intentions Structures: DIS (left) and DIS* (right)

2.2 Overview of the semantics of dynamic intention structures

The semantics of the intention structures defined in this article is based on their translation into a version of first-order modal logic developed by Fitting and Mendelsohn [8] having the following features.

- *The iota operator*: Terms of the form, $(\imath s.\phi(s))$, can be glossed as "the unique object s that satisfies $\phi(s)$." More precisely, if there is a unique object in the semantic domain such that the interpretation of ϕ holds for that object, then the iota expression designates that object; otherwise, it fails to designate. The iota operator is useful for constructing *definite descriptions* such as "the car that Zoe selects".
- *Predicate abstractions*: Predicate abstractions, which have the form, $\langle\lambda s.\ \phi(s)\rangle$, are similar to *lambda expressions* in the lambda calculus. The application of a predicate abstraction to the term t is notated: $\langle\lambda s.\ \phi(s)\rangle\ (t)$. If ϕ has no occurrences of modal operators, then this expression is equivalent to what would, in most logics, be written as $\phi(t)$. However, things get more interesting when ϕ involves a modal operator. For example, the term t is evaluated outside the scope of the modal $\Box$ operator in $\langle\lambda s.\ \Box\psi(s)\rangle\ (t)$, whereas it is evaluated inside the scope of the $\Box$ operator in $\Box(\langle\lambda s.\ \phi(s)\rangle(t))$.

Intention operators; and the Achieved *and* Done *predicates*

This article presumes the existence of a suitable intention operator and uses it in the logical formulas generated by the translation of intention structures.[3] The paper restricts attention to intentions having propositional content, such as "I intend that we go to the movies tonight." We employ expressions of the form, $\mathit{IntTh}(g, \phi)$, where *IntTh* is the intention operator, g is the agent holding the intention, and ϕ is the intended proposition.
Intention/plan identifiers. When agents coordinate their activities, it is useful to have identifiers for their plans. For example, if Alice and Bob have a plan to drive to Boston using some as yet unspecified car, then they might arbitrarily assign an identifier such as `Plan39` to their plan, thereby enabling them to refer to "the car in `Plan39`" instead of "the car that we plan to rent for our trip to Boston". Using identifiers in this way is especially convenient when the activities the agents are involved in might otherwise require lengthy descriptions to uniquely identify them. Similarly, in this article, we allow intentions to have identifiers associated with them. Thus, we presume that the intention operator can be augmented to include an additional *identifier* argument. For example, agent g's intention that ϕ, identified by id, would be represented by the formula, $\mathit{IntTh}(g, id, \phi)$.

[3] In this article, we do not provide a semantics for the intention operator. Instead, we focus on the use of DISs to provide a fine-grained structure to the content of an intention, with an eye toward its subsequent manipulation. In Sect. 7, we present an overview of a companion paper [27], in which we address the semantics of intention in detail.

Although one could always distinguish intentions by adding more descriptive content such as time or place, we include identifiers for convenience, to distinguish otherwise identical intentions and to simplify reference to them. However, not all intentions need have identifiers—for example, intentions that are inferred from others. To handle such cases one can simply add an axiom of the form, $\mathit{IntTh}(g, id, \phi) \Rightarrow \mathit{IntTh}(G, \phi)$, where the converse need not hold.

Act types. We presume that actions fall into types, called *act types*. Act types can be either *basic* or *complex*. A basic act type represents a class of atomic actions that, under the right circumstances, a capable agent can directly perform. A complex act type represents a class of actions that can be performed by executing a set of subsidiary tasks, commonly called a *recipe*. Thus, an agent might rent a car by walking to the rental car agency, filling out a form, paying lots of money, and so forth. Similarly, a group of agents might build a house by laying a foundation, buying some wood, nailing boards together, and so forth.

Following Ortiz [28], we allow act types to be partially specified using the @ constructor. An act type expression has the form, $A@\mathtt{arg}_1(val_1)\ldots@\mathtt{arg}_n(val_n)$, where A is an act type, each $\mathtt{arg}_i$ is the name of an argument, and each val_i is the value of the ith argument. For example, a `drive` act type, by itself, does not specify any arguments; however, `drive@obj(Car39)@to(Boston)` specifies a car and a destination.

Intending to do an action. Since all intentions in this aricle have propositional content, an agent g's intention to do an action α is represented as an intention that α is done by g. For this purpose, we employ the *Done* predicate, where $\mathit{Done}(g, \alpha)$ is true if g has done an action of type α. In many cases, the act type expression will include an `agt` argument that specifies the agent of the action. For example, `drive@agt(Bob)@obj(Car39)` specifies the act type of Bob driving a particular car. In such a case, the agent argument of the *Done* predicate is redundant and is therefore omitted. For example, the formula, $\mathit{Done}(\mathtt{drive@agt(Bob)@obj(Car39)})$, represents that `Bob` has done a `drive` action using `Car39`. Furthermore, Bob's intention to do the indicated `drive` action is represented as follows.

$$\mathit{intTh}(\mathtt{Bob}, \mathit{Done}(\mathtt{drive@agt(Bob)@obj(Car39)}))$$

Intentions in the context of collaborative activity. The SharedPlans formalization [14,15] specifies the mental state (i.e., intentions and beliefs) of a group of collaborating agents. In particular, when a group of agents are collaborating on a group activity α, then each agent in the group holds (among other things) an intention that can be glossed as "I intend that we do α." Grosz and Hunsberger [13] have identified a constraint they call the Coordinated Cultivation Requirement (CCR) that constrains a collaborating agent from certain forms of unilateral decision making. In their model of collaborative activity, called the Coordinated Cultivation of SharedPlans (CCSP) model, intentions subject to the CCR are called Group-Activity-Related (GAR) intentions. For the purposes of this article, the effect of the CCR on an agent's subsequent cultivation of its GAR intention is not important. However, a GAR intention does require an extra argument that specifies the group to which the collaborating agent belongs. In this article, we represent the GAR intention held by an agent g, in a group GR, toward a proposition ϕ, with plan identifier id as follows:

$$\mathit{IntTh}^*(g, id, GR, \phi)$$

In the case that the GAR intention concerns a group's doing of α, then it would have the following form:

$$\mathit{IntTh}^*(g, id, GR, \mathit{Done}(\alpha@\mathtt{agt}(GR)))$$

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

The achievement operator. In the context of collaborative activity, a group of agents might decide that some agent (or subgroup) g should achieve some subsidiary goal ϕ. In that case, the content of each group member's GAR intention would include a clause that can be glossed as "that g achieves ϕ." In this article, such clauses are represented using the *Achieved* operator, where an expression of the form, $Achieved(g, \phi)$, represents that the agent (or subgroup) g achieved the proposition ϕ.[4] Thus, an agent g_1 in a collaboration might hold the following GAR intention:

$$IntTh^*(g_1, id, GR, Achieved(g_2, \phi)).$$

2.3 Preliminary examples of intention structures

This section provides some preliminary examples of Dynamic Intention Structures (DISs). The examples illustrate the syntax of DISs and their translation into first-order logic. The syntax and semantics of DISs are formally defined in Sects. 3 and 4, respectively.

Example 1 Alice intends to rent a car

This example is interesting because Alice may intend to rent a car without having any particular car in mind yet. An important requirement is that Alice should have some way of referring to the car that she intends to rent (e.g., she may later declare that the car that she intends to rent will be blue) even if she has not yet selected any such car. Here's the DIS, D_1, for Alice's intention:[5]

D_1 =

ID/Agt:	id1 / A
ExVars:	$\mathtt{v}_1$
ActType:	rent@obj($\mathtt{v}_1$)
Conds:	$Car(\mathtt{v}_1)$

The constant id1 serves as an identifier for Alice's intention. Alice is represented by the constant A. The constant $\mathtt{v}_1$ is the parameter representing the car. The act type expression, rent@obj($\mathtt{v}_1$), represents the rental action. The constraint, $Car(\mathtt{v}_1)$, stipulates that the object to be rented must be a car. The translation of D_1 into first-order logic, which we notate as $\|D_1\|$, is given below.

$$\|D_1\| \;=\; IntTh(\mathtt{A}, \mathtt{id1}, (\exists x_1)(Done(\mathtt{rent@agt(A)@obj}(x_1)) \wedge Car(x_1)))$$

Notice that each occurrence of the parameter $\mathtt{v}_1$ in D_1 has been replaced by an occurrence of the existentially quantified variable x_1 in the generated formula, $\|D_1\|$. The existential quantification indicates that Alice intends that she rent some car, even if she does not yet know which car she will eventually rent. In addition, the act type expression from D_1 has been augmented to stipulate that Alice is the agent of the action. The augmented act type expression appears within the *Done* predicate in the generated formula.

In this example, Alice's selection of a value for the car parameter is not explicitly represented. If it is desired to do so, there are at least two alternatives: (1) represent the act of selecting a car; or (2) represent the goal of having selected a car. In this article, we focus on the latter alternative, employing a predicate, *Sel*, to represent an agent having selected an item to be the value of a parameter for a given plan.

[4] Saying that g achieved ϕ is not a statement about g's mental state. g might have achieved ϕ accidentally or intentionally.

[5] To save space, only the non-empty fields in a DIS are shown in the box notation.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

For example, consider D'_1, given below, which is identical to D_1, except that it has been augmented to include the proposition, $\mathit{Sel}(\mathtt{A}, \mathtt{v}_1, \text{“}\mathtt{v}_1\text{”}, \mathtt{id1})$, in the *Conds* field. The semantics of this extra entry is revealed by the translation of D'_1 into first-order logic.

$$D'_1 = \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{id1}\,/\,\mathtt{A} \\ \mathit{ExVars}: & \mathtt{v}_1 \\ \mathit{ActType}: & \mathtt{rent@obj(v_1)} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{v}_1), \mathit{Sel}(\mathtt{A}, \mathtt{v}_1, \text{“}\mathtt{v}_1\text{”}, \mathtt{id1}) \\ \hline \end{array}$$

$$\|D'_1\| = \mathit{IntTh}(\mathtt{A}, \mathtt{id1}, (\exists x_1)(\mathit{Done}(\mathtt{rent@agt(A)@obj}(x_1)) \wedge \mathit{Car}(x_1) \\ \wedge\, \mathit{Sel}(\mathtt{A}, x_1, \mathtt{v}_1, \mathtt{id1})))$$

Notice that each occurrence of the parameter $\mathtt{v}_1$ in D'_1—except for the quoted one—is translated into an occurrence of the existentially quantified variable x_1 in the generated formula. In contrast, the quoted term, “$\mathtt{v}_1$”, is translated into $\mathtt{v}_1$. Thus, $\mathtt{v}_1$ is translated into a term designating the object that `A` selects, whereas “$\mathtt{v}_1$” is translated into a constant designating the name of the parameter for which that object is selected. In the generated formula, $\mathit{Sel}(\mathtt{A}, x_1, \mathtt{v}_1, \mathtt{id1})$, holds if the item denoted by x_1 has been selected by `A` to be the value of the parameter named $\mathtt{v}_1$ in the plan identified by `id1`. For example, $\mathit{Sel}(\mathtt{A}, \mathtt{Car39}, \mathtt{v}_1, \mathtt{id1})$ holds if Alice has selected the car denoted by `Car39` to be the value of the parameter $\mathtt{v}_1$ in that plan.

The quoting of variables discussed above is related to the quoting of expressions in programming languages such as Lisp [34]. For example, in Lisp, an expression such as, `(let ((v 'x)) (cons v 'v))`, is syntactically valid. It evaluates to the list, `(x v)`. The reason is that `v` evaluates to `x`, and `'v` evaluates to `v`. In the same sense, the structure D'_1 contains the terms $\mathtt{v}_1$ and “$\mathtt{v}_1$”. The “evaluation” (or translation) of D'_1 yields a formula containing the corresponding terms, x_1 and $\mathtt{v}_1$. In this case, $\mathtt{v}_1$ translates into x_1, which is a logical variable denoting the car selected by the agent A; and “$\mathtt{v}_1$” translates into $\mathtt{v}_1$, which is a logical constant denoting a *name* that A can use to *refer* to that car—even before such a car has been selected.

Example 2 Alice intends to rent whatever car Zoe selects

This example is similar to the first example, except that Zoe will be selecting the car that Alice rents. Thus, in addition to Alice's intention to rent a car, we also consider Zoe's intention to select a car for Alice to rent.

Alice's intention. Alice's intention is given by a DIS—call it D_2—in which the car to be rented is represented by a *DefVar* specification—that is, a variable whose value is given by a *definite description.*

$$D_2 = \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{id2}\,/\,\mathtt{A} \\ \mathit{DefVars}: & (\mathtt{v}_2, \mathit{Sel}(\mathtt{Z}, \_, \text{“}\mathtt{v}_2\text{”}, \mathtt{id2})) \\ \mathit{ActType}: & \mathtt{rent@obj(v_2)} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{v}_2) \\ \hline \end{array}$$

The *DefVars* field normally contains a list of specifications. In this case, there is only one. It stipulates that the value of the parameter named $\mathtt{v}_2$ is to be whatever object satisfies the predication, $\mathit{Sel}(\mathtt{Z}, \_, \text{“}\mathtt{v}_2\text{”}, \mathtt{id2})$, where the underscore character is a place holder for the object in question. In other words, the value of the parameter $\mathtt{v}_2$ is to be whatever car Zoe selects. Notice that although Zoe will be making the selection, the parameter, $\mathtt{v}_2$, and the ID, `id2`, refer to elements of Alice's DIS, D_2.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

The translation of D_2 into first-order logic is given by:

$$\|D_2\| \;=\; \mathit{IntTh}(\mathtt{A}, \mathtt{id2}, \langle \lambda x_2.\, \Phi(x_2)\rangle\ \Upsilon),$$

where:

$$\begin{array}{lll} \Phi(x_2) & \equiv & \mathit{Done}(\mathtt{rent@agt(A)@obj(}x_2\mathtt{)}) \wedge \mathit{Car}(x_2) \\ \Upsilon & \equiv & (\imath s.\mathit{Sel}(\mathtt{Z}, s, \mathtt{v}_2, \mathtt{id2})). \end{array}$$

Notice that the *DefVar* parameter, $\mathtt{v}_2$, is translated into a variable, x_2, that is bound within a lambda expression in the generated formula. That lambda expression is applied to the value, $(\imath s.\mathit{Sel}(\mathtt{Z}, s, \mathtt{v}_2, \mathtt{id2}))$, which is a definite description involving the iota operator derived from the second part of the *DefVar* specification.[6] The iota expression can be glossed as "whatever object, s, that Zoe selects to be the value of the parameter, $\mathtt{v}_2$, in Alice's plan identified by id2." Hence we see an important property of DIS's: their incrementality. Whereas in ordinary logic the elements in the scope of a quantifier cannot be accessed outside the sentence, in DIS—as in DRT—they can be immediately accessed for further modification.

Zoe's intention. Zoe's intention that she select some car for Alice can be represented by the following DIS—call it $\hat{D}_2$—where the parameter named $\hat{\mathtt{v}}_2$ in $\hat{D}_2$ is distinct from the parameter named $\mathtt{v}_2$ in D_2.

$\hat{D}_2$ =

ID/Agt:	id2b / Z
ExVars:	$\hat{\mathtt{v}}_2$
Conds:	$\mathit{Car}(\hat{\mathtt{v}}_2) \wedge \mathit{Sel}(\mathtt{Z}, \hat{\mathtt{v}}_2, \text{“}\mathtt{v}_2\text{”}, \mathtt{id2})$

$$\|\hat{D}_2\| \;=\; \mathit{IntTh}(\mathtt{Z}, \mathtt{id2b}, (\exists \hat{x}_2)(\mathit{Car}(\hat{x}_2) \wedge \mathit{Sel}(\mathtt{Z}, \hat{x}_2, \mathtt{v}_2, \mathtt{id2})))$$

Notice that Zoe intends to select some car $\hat{x}_2$ to be the value of Alice's parameter $\mathtt{v}_2$ in her plan identified by id2. In addition, the semantics of Zoe's intention ensures that $\hat{x}_2$ will also be the value of Zoe's parameter $\hat{\mathtt{v}}_2$, since $\hat{\mathtt{v}}_2$ gets translated into $\hat{x}_2$.

2.3.1 Intentions in the context of collaborative group activity

Example 2 explored the related intentions of Alice and Zoe; however, Alice and Zoe did not have a SharedPlan. Thus, there were no GAR intentions [13]. The next examples consider GAR intentions in the context of collaborative activity. In our model, a GAR intention is represented by a DIS*, which has an extra field denoting the group to which the intending agent belongs. The examples also explore the *SubBoxes* field of a DIS. (Although the *SubBoxes* field can be used in the context of single-agent activity; its most interesting features arise in the context of collaborative group activity.)

Example 3 Alice and Bob plan to travel to boston together

In this example, Alice and Bob each hold a GAR intention concerning their plan. The contents of their GAR intentions are identical, except for the *Agt* field. Let GR denote the group consisting of Alice and Bob; and let g denote either Alice (A) or Bob (B). Then D_3, below, is a DIS* that represents g's GAR intention.[7]

D_3 =

ID/Agt /Grp:	id3 / g / GR
ActType:	travel@to(Boston)

[6] The choice of s in the iota expression is arbitrary. It replaces the underscore from the *DefVar* entry.

[7] Recall that only non-empty fields are shown in the box notation.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

$$\|D_3\| \;=\; \mathit{IntTh}^*(g, \texttt{id3}, \texttt{GR}, \mathit{Done}(\texttt{travel@agt(GR)@to(Boston)}))$$

Next, we consider a slight modification of this example in which the destination is represented by a variable instead of a constant. The idea here is to facilitate replanning should it turn out that the group will be unable to go to Boston. In that case, they might decide to change the value of their destination variable to New York. The modified DIS*, called D'_3, is defined below. In this case, a *DefVar* is used to represent the destination of the trip. The value of this *DefVar* is represented by the constant Boston.

$$D'_3 \;=\; \begin{array}{|ll|}\hline \mathit{ID/Agt/Grp}: & \texttt{id3} \,/\, g \,/\, \texttt{GR} \\ \mathit{DefVars}: & (\texttt{v}_3, \texttt{Boston}) \\ \mathit{ActType}: & \texttt{travel@to}(\texttt{v}_3) \\ \hline \end{array}$$

$$\|D'_3\| \;=\; \mathit{IntTh}^*(g, \texttt{id3}, \texttt{GR}, \langle \lambda x_3.\, \mathit{Done}(\texttt{travel@agt(GR)@to}(x_3))\rangle \texttt{Boston})$$

Example 4 Alice and Bob intend that Alice should do β_1 subject to the constraint ϕ_1 and Bob should do β_2 subject to the constraint ϕ_2

In this example, we suppose that Alice and Bob begin by making a group decision (or agreement) that Alice shall do β_1 subject to the constraint ϕ_1 and Bob shall do β_2 subject to the constraint ϕ_2. According to the CCSP model [13], such an agreement entails certain obligations on the participants.[8] First, it obliges Alice and Bob to each adopt a GAR intention concerning their group activity. Second, it obliges Alice to adopt a subsidiary intention concerning her doing of β_1, and Bob to adopt a subsidiary intention concerning his doing of β_2. The following discussion concentrates on the representation of these intentions. For expository convenience, the subsidiary intentions are addressed first.

Alice and Bob's subsidiary intentions. Alice's subsidiary intention concerning her doing of β_1 can be represented by the DIS, D_{4a}, defined below (on the left). Similarly, Bob's subsidiary intention concerning his doing of β_2 can be represented by the DIS, D_{4b}, defined below (on the right).

$$D_{4a} \;=\; \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \texttt{id4a} \,/\, \texttt{A} \\ \mathit{ActType}: & \beta_1 \\ \mathit{Conds}: & \phi_1 \\ \hline \end{array} \qquad D_{4b} \;=\; \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \texttt{id4b} \,/\, \texttt{B} \\ \mathit{ActType}: & \beta_2 \\ \mathit{Conds}: & \phi_1 \\ \hline \end{array}$$

$$\|D_{4a}\| \;=\; \mathit{IntTh}(\texttt{A},\ \texttt{id4a},\ \phi_1 \wedge \mathit{Done}(\beta_1\texttt{@agt(A)})) \qquad \|D_{4b}\| \;=\; \mathit{IntTh}(\texttt{B},\ \texttt{id4b},\ \phi_2 \wedge \mathit{Done}(\beta_2\texttt{@agt(B)}))$$

Alice and Bob's GAR intentions. The content of each agent's GAR intention refers *both* to Alice's doing of β_1 and Bob's doing of β_2. Their GAR intentions, represented by D_4 below, can be glossed as "I intend that Alice do β_1 subject to ϕ_1 and Bob do β_2 subject to ϕ_2." Once again, g represents Alice or Bob.

$$D_4 \;=\; \begin{array}{|ll|}\hline \mathit{ID/Agt/Grp}: & \texttt{id4} \,/\, g \,/\, \texttt{GR} \\ \mathit{SubBoxes}: & \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \texttt{id4a} \,/\, \texttt{A} \\ \mathit{ActType}: & \beta_1 \\ \mathit{Conds}: & \phi_1 \\ \hline \end{array} \, , \, \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \texttt{id4b} \,/\, \texttt{B} \\ \mathit{ActType}: & \beta_2 \\ \mathit{Conds}: & \phi_2 \\ \hline \end{array} \\ \hline \end{array}$$

Notice that D_4 contains two subsidiary boxes that are identical to D_{4a} and D_{4b} given above. Based on the description of the translation function given so far, the translation of D_4 into

[8] As discussed by Grosz and Hunsberger [13], the obligations entailed by such agreements have been pointed out by many philosphers, including Bratman [2], Gilbert [10] and Tuomela [35].

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

first-order logic would generate a formula containing nested intentions. Such a formula might be glossed as, "I intend that Alice intend that she does $\beta_1 \ldots$". However, this does not capture the desired relationship between the GAR intention and the subsidiary activities: "I intend that Alice *do* $\beta_1 \ldots$".[9] To distinguish the "stand-alone" translation of a DIS—such as the stand-alone translations of D_{4a} and D_{4b} given earlier—and the "in-context" translation of a DIS as a subsidiary box within a parent DIS, we employ two different translation functions. The *stand-alone* translation function, which is the only one that has been seen so far, generates formulas involving the intention operator. The *in-context* translation function, which has not yet been seen, generates formulas involving the *Achieved* operator. For a DIS such as D_4, which contains subsidiary boxes, the translation of everything but its subsidiary boxes is performed by the stand-alone translation function, which generates the top-level intention formula. In contrast, the translation of its subsidiary boxes is performed by the in-context translation function, which generates subsidiary formulas, Φ_1 and Φ_2, involving the *Achieved* operator.

$$\|D_4\| \;=\; \mathit{IntTh}^*(g, \mathtt{id4}, \mathtt{GR}, \Phi_1 \wedge \Phi_2),$$

where:

$$\begin{aligned} \Phi_1 &\equiv \mathit{Achieved}(\mathtt{A}, \phi_1 \wedge \mathit{Done}(\beta_1 @ \mathtt{agt}(\mathtt{A}))) \\ \Phi_2 &\equiv \mathit{Achieved}(\mathtt{B}, \phi_2 \wedge \mathit{Done}(\beta_2 @ \mathtt{agt}(\mathtt{B}))) \end{aligned}$$

The stand-alone and in-context translation functions are formally defined in Sect. 4.

Example 5 Alice and Bob plan to drive whatever car Alice selects

In this example, Alice and Bob have a SharedPlan to drive whatever car Alice selects. (Here, the `drive` act type represents a multi-agent action that Alice and Bob will do together.) Thus, Alice and Bob each have a GAR intention to that effect. In addition, Alice has an intention concerning her selection of a car.

The GAR intentions. The DIS*, D_5, represents the GAR intention held by g (either Alice or Bob).

$$D_5 \;=\; \boxed{\begin{array}{ll} \mathit{ID/Agt/Grp}: & \mathtt{id5} \,/\, g \,/\, \mathtt{GR} \\ \mathit{DefVars}: & (\mathtt{v}_5, \{\mathit{Sel}(\mathtt{A}, \_, \text{"}\mathtt{v}_5\text{"}, \mathtt{id5}), \mathit{Car}(\_)\}) \\ \mathit{ActType}: & \mathtt{drive@obj(v_5)} \\ \mathit{SubBoxes}: & \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \mathtt{id5a} \,/\, \mathtt{A} \\ \mathit{ExVars}: & \mathtt{v}_6 \\ \mathit{Conds}: & \mathit{Sel}(\mathtt{A}, \mathtt{v}_6, \text{"}\mathtt{v}_5\text{"}, \mathtt{id5}) \wedge \mathit{Car}(\mathtt{v}_6) \end{array}} \end{array}}$$

$$\|D_5\| \;=\; \mathit{IntTh}^*(g, \mathtt{id5}, \mathtt{GR}, \langle \lambda x_5.\, \Psi_1 \wedge \Psi_2 \rangle\; (\iota s.\mathit{Sel}(\mathtt{A}, s, \mathtt{v}_5, \mathtt{id5}) \wedge \mathit{Car}(s))),$$

$$\begin{aligned} \text{where:}\quad \Psi_1 &\equiv \mathit{Done}(\mathtt{drive@agt(GR)@obj}(x_5)) \\ \Psi_2 &\equiv \mathit{Achieved}(\mathtt{A}, (\exists x_6)(\mathit{Sel}(\mathtt{A}, x_6, \mathtt{v}_5, \mathtt{id5}) \wedge \mathit{Car}(x_6))) \end{aligned}$$

Notice that the *DefVar* specification for $\mathtt{v}_5$ is a set of predicates containing the underscore placeholder. These predicates are conjoined within the corresponding iota expression in the translation, $\|D_5\|$. In addition, notice that the "in context" translation of a subsidiary box yields an *Achieved*$(\ldots)$ clause.

Alice's intention concerning her selection of a car. Alice's intention concerning her selection of a car is represented by a DIS—call it D_6—that is identical to the subsidiary box in

[9] Grosz and Hunsberger [13] discuss the relationship between GAR intentions and subsidiary activities in more detail.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

D_5 above. However, the stand-alone translation of D_6 yields an intention formula, not an *Achieved* formula. In particular, A intends that there is a car x_6 that she selects as the value of $\mathtt{v}_5$ in the DIS with id id5.

$$D_6 = \begin{array}{|ll|}\hline \textit{ID/Agt}: & \mathtt{id5a}\,/\,\mathtt{A} \\ \textit{ExVars}: & \mathtt{v}_6 \\ \textit{Conds}: & \textit{Sel}(\mathtt{A}, \mathtt{v}_6, \text{“}\mathtt{v}_5\text{”}, \mathtt{id5}) \wedge \textit{Car}(\mathtt{v}_6) \\ \hline \end{array}$$

$$\|D_6\| = \textit{IntTh}(\mathtt{A}, \mathtt{id5a}, (\exists x_6)(\textit{Sel}(\mathtt{A}, x_6, \mathtt{v}_5, \mathtt{id5}) \wedge \textit{Car}(x_6)))$$

It is important to point out that the parameters, $\mathtt{v}_5$ (in D_5) and $\mathtt{v}_6$ (in D_6), refer to the *same car* in the following sense. First, the only way that Alice can satisfy her intention D_6 is by ensuring that there is some car—call it X—for which the predicate $\textit{Sel}(\mathtt{A}, X, \mathtt{v}_5, \mathtt{id5})$ holds. (Alice would normally establish such a predication by declaring that she has made such a selection.) In such a case, the value of her parameter $\mathtt{v}_6$, which corresponds to the existentially quantified variable x_6 in the translation of D_6, would be X. Moreover, the iota expression, $(\iota s.\textit{Sel}(\mathtt{A}, s, \mathtt{v}_5, \mathtt{id5}) \wedge \textit{Car}(s))$, in the formula representing the translation of D_5 would also denote X. Thus, the only way the GAR intention represented by D_5 could be satisfied would be if the value of $\mathtt{v}_5$, which corresponds to the lambda variable x_6 in the translation of D_5, were also X.

3 The syntax of dynamic intention structures

This section presents the syntax of DIS structures, which will be slightly extended in Sect. 6.2.

First, we presume the following sets of symbols:

- *IdNames*, *AgtNames* and *GrpNames*—sets of constant symbols used for plan/intention identifiers (e.g., id1, id2, id3, etc.), agent names (e.g., A, B, C, etc.) and names of groups of agents (e.g., GR)
- *Constants*—a set of constant symbols that includes the above sets as subsets, but may include other symbols as well (e.g., Boston, Car39 and Chair61)
- *VarNames*—a set of symbols used for *ExVar* and *DefVar* parameter names (e.g., v, $\mathtt{v}_1$, w, etc.)
- *ActTypeNames*—a set of symbols used as act-type names (e.g., rent, travel and drive)
- *ActTypeArgs*—a set of symbols used as names of arguments within act type expressions (e.g., agt, from, to and obj)
- *PredNames*—a set of predicate symbols (e.g., *Blue* or *Econ*) that does *not* include the symbol *Sel*

A DIS contains the following fields: *ID*, *Agt*, *ExVars*, *DefVars*, *ActType*, *SubBoxes* and *Conds*. A DIS* contains all of these fields together with a field named *Grp*. The contents of these fields are constrained as follows. Each item is numbered to faciliate future reference.

(1) The *ID* field must contain a constant, $id \in \textit{IdNames}$.
(2) The *Agt* field must contain a constant, $g \in \textit{AgtNames}$.
(3) The *Grp* field must contain a constant, $GR \in \textit{GrpNames}$.
(4) The *ExVars* field must contain a list of zero or more constants belonging to the set *VarNames*.
(5) The *DefVars* field must contain a list of zero or more *DefVar* specifications, each having one of the following two forms:

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

(a) (v, X), where:
- $v \in VarNames$, but $v \notin ExVars$; and
- X can be a constant, $c \in Constants$; or a set of zero or more propositions, each having the form, $\phi(\_, t_1, \ldots, t_n)$, where $\phi \in PredNames$, $n \geq 0$, and each $t_i \in Constants \cup VarNames$.

(b) (v, Y, g_2, id_2, v_2), where:
- $v \in VarNames$, but $v \notin ExVars$;
- Y is a set of propositions, like X above, except that exactly one of the propositions in Y must have the form, $Sel(g_2, \_, \text{"}v\text{"}, id)$, where g_2 is as described below and id is the value of the *ID* field for this DIS.[10]
- $g_2 \in AgtNames \cup GrpNames$;
- $id_2 \in IdNames$; and
- $v_2 \in VarNames$.

Note: The variables appearing as first arguments of *DefVar* specifications must be distinct.

(6) The *ActType* field must be empty or contain a single act-type expression of the form $type@arg_1(val_1) \ldots @arg_n(val_n)$, where $type \in ActTypeNames$, $\{arg_1, \ldots, arg_n\}$ is an n-element subset of *ActTypeArgs*, and each $val_i \in Constants \cup VarNames$;

(7) The *SubBoxes* field in a DIS structure must contain a set of zero or more DIS structures; the *SubBoxes* field in a DIS* structure can contain either DIS or DIS* structures (or both).

(8) The *Conds* field must contain a set of zero or more conditions, each having one of the following forms:
- $\phi(t_1, \ldots, t_n)$, where $\phi \in PredNames$ and each $t_i \in VarNames \cup Constants$; or
- $Sel(g, v, \text{"}w\text{"}, id)$, where g is the value of the *Agt* field for this DIS, $v \in VarNames$, $w \in VarNames$, and $id \in IdNames$.

4 The semantics of dynamic intention structures

As has already been seen, there are two ways that a DIS can be translated into a formula in first-order logic: either as a stand-alone box, which yields an intention formula, or as a subsidiary box (i.e., in the context of a parent structure), which yields an $Achieved(\ldots)$ formula. Each translation function takes a DIS or DIS* structure, D, as its input and generates as its output a sentence in first-order modal logic. $\|D\|$ is the translation of D as a stand-alone box, while $\|D\|^c$ is the "in context" translation of D as a subsidiary box.

A note on logical combinations of intentions. In this article we restrict attention to DISs whose content involves conjunctions of propositions within the scope of various quantifiers. In the companion paper [27] on intention revision, we address arbitrary logical combinations of intentions, including conditional intentions, and their consequences.

A note about free variables. The syntax rules presented in Sect. 3 permit DISs to have free variables. For example, a DIS might have the proposition, $Car(\mathrm{v})$, in its *Conds* field, but not have any corresponding entry for v in its *ExVar* or *DefVar* fields. However, the translation functions defined in this section, like all of the examples seen so far in this article, restrict attention to DISs that do not have any free variables. Later on, in Sect. 6.2, examples involv-

[10] The last three arguments, g_2, id_2 and v_2, are optional. They are allowed only for bookkeeping convenience when some other agent or group, g_2, shall determine the value for v. Example 3 in Sect. 6.1 illustrates their use.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

ing free variables are considered. At that time, the definitions of the translation functions are extended to accommodate free variables.

A note about the optional arguments in a DefVar *specification.* The optional arguments in a *DefVar* specification (cf. item 5b in Sect. 3) are for convenience only. They play no role in the translation of a DIS or DIS* into a formula of first-order logic. Thus, they are not addressed in this section.

The most general forms of a DIS D (on the left) and a DIS* D^* (on the right) are shown below.

ID/Agt:	*id / g*
ExVars:	$\mathtt{v}_1, \ldots, \mathtt{v}_m$
DefVars:	$(\mathtt{w}_1, X_1), \ldots, (\mathtt{w}_n, X_n)$
ActType:	α
SubBoxes:	$\mathcal{S}_1, \ldots, \mathcal{S}_k$
Conds:	$\phi_1, \ldots, \phi_p$

ID/Agt/Grp:	*id / g / GR*
ExVars:	$\mathtt{v}_1, \ldots, \mathtt{v}_m$
DefVars:	$(\mathtt{w}_1, X_1), \ldots, (\mathtt{w}_n, X_n)$
ActType:	α
SubBoxes:	$\mathcal{S}_1, \ldots, \mathcal{S}_k$
Conds:	$\phi_1, \ldots, \phi_p$

"Stand-Alone" translation. The "stand-alone" translations of D and D^* are defined as follows.

$$\|D\| =_{def} IntTh(g, id, \langle \lambda y_1.\ \langle \lambda y_2.\ \cdots \langle \lambda y_n.\ \Omega(g)\rangle \Upsilon_n \ldots \rangle \Upsilon_2 \rangle \Upsilon_1)$$

$$\|D^*\| =_{def} IntTh^*(g, id, GR, \langle \lambda y_1.\ \langle \lambda y_2.\ \cdots \langle \lambda y_n.\ \Omega(GR)\rangle \Upsilon_n \ldots \rangle \Upsilon_2 \rangle \Upsilon_1)$$

where:

- $\Omega(A) \equiv (\exists x_1, \ldots, x_m)\ (\|\phi_1\| \wedge \cdots \wedge \|\phi_p\| \wedge Done(\|\alpha@\mathtt{agt}(A)\|) \wedge \|\mathcal{S}_1\|^c \wedge \cdots \wedge \|\mathcal{S}_k\|^c)$
- for each $j \in \{1, \ldots, p\}$, $\|\phi_j\|$ is the same as ϕ_j, except that all occurrences of $\mathtt{v}_i$, $\mathtt{w}_i$ and "$\mathtt{v}_i$" have been replaced by x_i, y_i and $\mathtt{v}_i$, respectively.
- $\|\alpha@\mathtt{agt}(A)\|$ is the same as $\alpha@\mathtt{agt}(A)$, except that all occurrences of $\mathtt{v}_i$ and $\mathtt{w}_i$ have been replaced by x_i and y_i, respectively;
- for each $e \in \{1, \ldots, n\}$, X_e can be either a constant symbol or a set of propositions.
  - If X_e is a constant symbol, c, then Υ_e is simply c.
  - If X_e is a set of propositions, $\{\Phi_1, \ldots, \Phi_{q_e}\}$, then Υ_e is the iota expression, $(\iota s_e \cdot \bigwedge_{j=1}^{q_e} \|\Phi_j\|)$, where each $\|\Phi_j\|$ depends on the form of Φ_j, as follows.
    - If Φ_j has the form, $\phi(\_, t_1, \ldots, t_r)$, then $\|\Phi_j\|$ is the same as $\phi(s_e, t_1, \ldots, t_r)$, except that all occurrences of $\mathtt{v}_i$ and $\mathtt{w}_i$ have been replaced by x_i and y_i, respectively.
    - If Φ_j has the form $Sel(g, \_, \text{"}v'\text{"}, id')$, then $\|\Phi_j\| \equiv Sel(g, e, v', id')$.

Notice that each subsidiary box, $\mathcal{S}_i$, is translated using the "in context" translation function, $\| \cdot \|^c$.

"In Context" translation. The "in context" translations of D and D^* are defined as follows.

$$\|D\|^c =_{def} \langle \lambda y_1.\ \langle \lambda y_2.\ \cdots \langle \lambda y_n.\ Achieved(g, \Omega(g))\rangle \Upsilon_n \ldots \rangle \Upsilon_2 \rangle \Upsilon_1$$

$$\|D^*\|^c =_{def} \langle \lambda y_1.\ \langle \lambda y_2.\ \cdots \langle \lambda y_n.\ Achieved(GR, \Omega(GR))\rangle \Upsilon_n \ldots \rangle \Upsilon_2 \rangle \Upsilon_1$$

where Ω and $\Upsilon_1, \ldots, \Upsilon_n$ area as given above. Notice that when treating D (resp. D^*) as a subsidiary box in the context of some parent box, its sub-boxes are also translated "in context", yielding $Achieved(\ldots)$ clauses.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

5 DIS-creation and DIS-update operations

This section presents the primitive DIS-creation and DIS-update operations that are used to create new DIS structures and incrementally update existing ones. The DIS-update operations discussed in this section are restricted to cases where new data is *added* to an existing DIS; they do not address cases where existing content is *deleted* from a DIS.[11] Furthermore, the DIS resulting from an update operation is not guaranteed to be self-consistent. For example, it is possible, although irrational, for an agent to add the condition "not red" to its intention to rent a red car. However, such cases are easily discovered by examining the translation of the resulting DIS into FOL. Finally, even if a new or updated DIS is self-consistent, it may not be consistent with the rest of the intentions in an agent's database of intentions. Such problems are resolved through a process of intention *revision*, which is treated in a companion paper [27].

In view of these restrictions, the rest of this article presumes that DIS-update operations are applied to consistent DISs, and that the resulting structures are also consistent.

Since the syntax of these operations is closely related to the syntax of the DIS structures presented in Sect. 3, the descriptions of most of the operations avoid needless repetition by referring to the corresponding numbered items from Sect. 3.

5.1 DIS-creation operations

There are two DIS-creation operations.

- $NewDIS(id, g)$—creates a new DIS with its *ID* field set to $id \in IdNames$ and its *Agt* field set to $g \in AgtNames$ (cf. items 1 and 2 in Sect. 3).
- $NewDIS^*(id, g, GR)$—creates a new DIS* with *ID* set to $id \in IdNames$, *Agt* set to $g \in AgtNames$, and *Grp* set to $GR \in GrpNames$ (cf. items 1, 2 and 3 in Sect. 3).

5.2 DIS-update operations

Each of the following DIS-update operations takes an existing DIS (or DIS*) structure as its first argument. In the descriptions below, D stands for an existing DIS (or DIS*) structure. Each DIS-update operation generates an updated version of D, which is guaranteed to be another DIS (or DIS*). Alternatively, each update operation can be viewed as destructively modifying the contents of its input DIS.

- $AddExVar(D, v)$—Adds the parameter $v \in VarNames$ to the *ExVar* field in D (cf. item 4 in Sect. 3). Only applicable if v does not already appear as an *ExVar* or *DefVar* parameter in D.
- $AddDefVar(D, v, X)$—Adds the entry (v, X) to the *DefVars* field in D, where (v, X) is as described in item 5a of Sect. 3. Not applicable if v already appears as an *ExVar* or *DefVar* parameter in D.
- $AddDefVar(D, \mathrm{v}, Y, g_2, id_2, v_2)$—Adds the entry (v, Y, g_2, id_2, v_2) to the *DefVars* field in D, where (v, Y, g_2, id_2, v_2) is as described in item 5b of Sect. 3. Not applicable if v already appears as an *ExVar* or *DefVar* parameter in D.

[11] That is not to say that deleting information from a DIS is not important. For example, I might originally intend to rent a red car, only to find out later that red cars are way too expensive for my budget. In response, I would normally delete the "red" condition from my intention. However, the deletion of content from a DIS is beyond the scope of this article.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

- *AddDefVarCond*(D, v, Φ)—Adds the condition, Φ, to the *DefVar* entry for v in D, where Φ has the form, $\phi(\_, t_1, \ldots, t_n)$, as described in item 5a of Sect. 3. Only applicable if D already contains a *DefVar* entry for v, as described in item 5b of Sect. 3.
- *AddActType*(D, α)—Adds the act type expression α to the *ActType* field of D (cf. item 6 in Sect. 3). Only applicable if the *ActType* field of D is empty.
- *AddActArg*$(D, @arg(val))$—Appends the act type argument, $@arg(val)$, to the *ActType* entry in D. Only applicable if the *ActType* field of D is non-empty and does not already contain an act-type argument named *arg* (cf. item 6 in Sect. 3).
- *AddSubBox*(D, D_s)—Adds D_s to the *SubBoxes* field of D. If D is a DIS, then D_s must be a DIS; if D is a DIS*, D_s can be either a DIS or a DIS* (cf. item 7 in Sect. 3).
- *AddConds*$(D, \{\Phi_1, \ldots, \Phi_p\})$—Adds the conditions, $\Phi_1, \ldots, \Phi_p$, to the *Conds* field in D, where each Φ_i must have one of the forms, $\phi(t_1, \ldots, t_n)$ or $Sel(g, v, \text{“}w\text{”}, id)$, as described in item 8 of Sect. 3.
- *ShiftVar*(D, v, X)—Removes the parameter v from the *ExVar* field of D, and adds the entry, (v, X), to the *DefVar* field of D, where (v, X) is as described in item 5a of Sect. 3. Only applicable if D already has v as an *ExVar* entry.
- *ShiftVar*$(D, v, Y, g_2, id_2, v_2)$—Removes the parameter v from the *ExVar* field of D, and adds the entry, (v, Y, g_2, id_2, v_2), to the *DefVar* field of D, where (v, Y, g_2, id_2, v_2) is as described in item 5b of Sect. 3. Only applicable if D already has v as an *ExVar* entry.

6 Sample scenario illustrating dynamic intention structures

This section presents a dynamic scenario involving four agents—Alice, Bob, Chris and Zoe—represented as `A`, `B`, `C` and `Z`, respectively. Alice, Bob and Chris constitute a group, `GR`, planning to travel to Boston together. The scenario is dynamic because, we presume, the GAR intentions held by the agents motivate them to participate in group decision-making processes aimed at elaborating their partially specified plan. When they successfully arive at a group decision (e.g., that Alice should be the one to get the car), it obliges them to update their GAR intentions and, in some cases, to adopt new intentions. Although the examples in this section may make reference to the decisions that prompted the agents to adopt new intentions or update existing intentions, the focus here is on representing the intentions themselves using DIS structures and on the operations that create new DISs or update existing DISs.[12]

In the process of elaborating their plan, the group decides that Alice should be the one to get the car. In response, Alice subsequently adopts a separate intention to rent a car from Zoe. Since this side scenario is simpler, we begin with it.

6.1 Side scenario: Alice rents a car

For the sake of expositional simplicity, we assume that Alice and Zoe do not have all of the intentions required of a SharedPlan. In particular, they have no GAR intentions. Instead, each simply has an intention concerning certain aspects of the rental action. The main scenario, treated afterward, examines intentions (including GAR intentions) in the context of a SharedPlan.

[12] Grosz and Hunsberger [13] discuss in detail the relationships between GAR intentions, group decision-making processes, group decisions, obligations and intention updates.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

Example 1 Alice intends to rent a car

This example has already been encountered (cf. Example 1 in Sect. 2). However, here we provide some additional details. Alice's intention to rent a car can be represented by a DIS. The creation of that DIS can be decomposed into several primitive operations—namely, to create a new DIS and then update it. The DIS—call it D_A—representing Alice's intention to rent a car is created by the following operations.

[$\sigma_{1.1}$] $D_A = \mathit{NewDIS}(\texttt{idA}, \texttt{A})$—Create a new DIS with id `idA` for the agent `A`.
[$\sigma_{1.2}$] $\mathit{AddExVar}(D_A, \texttt{w}_\texttt{c})$—Add a new *ExVar* parameter named $\texttt{w}_\texttt{c}$ to D_A.
[$\sigma_{1.3}$] $\mathit{AddConds}(D_A, \{\mathit{Car}(\texttt{w}_\texttt{c})\})$—Add the condition, $\mathit{Car}(\texttt{w}_\texttt{c})$, to D_A.
[$\sigma_{1.4}$] $\mathit{AddActType}(D_A, \texttt{rent@obj}(\texttt{w}_\texttt{c}))$—Add the act-type, $\texttt{rent@obj}(\texttt{w}_\texttt{c})$, to D_A.

Here is the resulting DIS and its translation into first-order logic:

$$D_A = \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \texttt{idA / A} \\ \mathit{ExVars}: & \texttt{w}_\texttt{c} \\ \mathit{ActType}: & \texttt{rent@obj}(\texttt{w}_\texttt{c}) \\ \mathit{Conds}: & \mathit{Car}(\texttt{w}_\texttt{c}) \end{array}}$$

$$\|D_A\| = \mathit{IntTh}(\texttt{A}, \texttt{idA}, (\exists y_c)(\mathit{Done}(\texttt{rent@agt(A)@obj}(y_c)) \wedge \mathit{Car}(y_c)))$$

Example 2 Alice decides that the car she rents should be Blue

Alice's decision in this case is, in effect, a commitment to perform a single primitive update operation, $\sigma_{2.1}$.

[$\sigma_{2.1}$] $\mathit{AddConds}(D_A, \{\mathit{Blue}(\texttt{w}_\texttt{c})\})$—Add the condition, $\mathit{Blue}(\texttt{w}_\texttt{c})$, to D_A.

Here is the updated D_A and its translation into first-order logic:

$$D_A = \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \texttt{idA / A} \\ \mathit{ExVars}: & \texttt{w}_\texttt{c} \\ \mathit{ActType}: & \texttt{rent@obj}(\texttt{w}_\texttt{c}) \\ \mathit{Conds}: & \mathit{Car}(\texttt{w}_\texttt{c}), \mathit{Blue}(\texttt{w}_\texttt{c}) \end{array}}$$

$$\|D_A\| = \mathit{IntTh}(\texttt{A}, \texttt{idA}, (\exists y_c)(\mathit{Done}(\texttt{rent@agt(A)@obj}(y_c)) \wedge \mathit{Car}(y_c) \wedge \mathit{Blue}(y_c)))$$

Example 3 Alice decides to rent the car from Zoe

This situation might arise, for example, if Alice (the customer) and Zoe (the rental car agent) made an agreement stipulating that Zoe would select a car for Alice, and Alice would rent that car. (Some features of this example are similar to those seen earlier in Example 5 of Sect. 2.) Such an agreement would entail obligations on Alice and Zoe. In particular, Alice would be obliged to rent whatever car Zoe selects, and Zoe would be obliged to select a car for Alice. In response, Alice updates her intention to reflect that Zoe will be responsible for selecting the car, and Zoe adopts an intention to select a car for Alice. (These responses could occur simultaneously or in either order.)
Alice's intention update. Alice updates her intention by applying the update operations $\sigma_{3.1}$ through $\sigma_{3.6}$, listed below, to her pre-existing DIS. Notice that part of Alice's updated intention stipulates that Zoe shall select a blue car for her. This part of Alice's intention is represented by a subsidiary box, D_s, generated by the update operations $\sigma_{3.3}$ through $\sigma_{3.6}$.

[$\sigma_{3.1}$] $\mathit{AddActArg}(D_A, \texttt{@from(Z)})$—Add the argument, `@from(Z)`, to the act type in D_A.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

[$\sigma_{3.2}$] $\mathit{ShiftVar}(D_A, \mathtt{w_c}, \{\mathit{Sel}(\mathtt{Z}, \_, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), \mathit{Car}(\_), \mathit{Blue}(\_)\}, \mathtt{Z}, \mathtt{idZ}, \mathtt{u_c})$
— Shift the variable associated with $\mathtt{w_c}$ in D_A from the *ExVar* category to the *DefVar* category. The value of this variable is to be determined by the given set of conditions. The arguments, $\mathtt{Z}$, $\mathtt{idZ}$ and $\mathtt{u_c}$, indicate that Zoe shall be selecting the value of $\mathtt{w_c}$, that the ID of Zoe's DIS is $\mathtt{idZ}$, and the parameter name for the car within Zoe's DIS is $\mathtt{u_c}$.[13]

[$\sigma_{3.3}$] $D_s = \mathit{NewDIS}(\mathtt{idZ}, \mathtt{Z})$—Create the new box, D_s.

[$\sigma_{3.4}$] $\mathit{AddExVar}(D_s, \mathtt{u_c})$—Add a new *ExVar* parameter named $\mathtt{u_c}$ to D_s.

[$\sigma_{3.5}$] $\mathit{AddConds}(D_s, \{\mathit{Sel}(\mathtt{Z}, \mathtt{u_c}, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), \mathit{Car}(\mathtt{u_c}), \mathit{Blue}(\mathtt{u_c})\})$
— Add the conditions, $\mathit{Sel}(\mathtt{Z}, \mathtt{u_c}, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA})$, $\mathit{Car}(\mathtt{u_c})$ and $\mathit{Blue}(\mathtt{u_c})$, to the box, D_s.

[$\sigma_{3.6}$] $\mathit{AddSubBox}(D_A, D_s)$—Add D_s as a subsidiary box to D_A.

Here is Alice's updated DIS and its translation into first-order logic.

$$D_A = \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{idA\,/\,A} \\ \mathit{DefVars}: & (\mathtt{w_c}, \{\mathit{Sel}(\mathtt{Z}, \_, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), \mathit{Car}(\_), \mathit{Blue}(\_)\}, \mathtt{Z}, \mathtt{idZ}, \mathtt{u_c}) \\ \mathit{ActType}: & \mathtt{rent@obj(w_c)@from(Z)} \\ \mathit{SubBoxes}: & \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{idZ\,/\,Z} \\ \mathit{ExVars}: & \mathtt{u_c} \\ \mathit{Conds}: & \mathit{Sel}(\mathtt{Z}, \mathtt{u_c}, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), \mathit{Car}(\mathtt{u_c}), \mathit{Blue}(\mathtt{u_c}) \\ \hline \end{array} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{w_c}), \mathit{Blue}(\mathtt{w_c}) \\ \hline \end{array}$$

$$\|D_A\| = \mathit{IntTh}(\mathtt{A}, \mathtt{idA}, \langle \lambda y_c.\ \Phi_1 \wedge \Phi_2 \rangle\ \Upsilon)$$

where:

$$\Phi_1 \equiv \mathit{Done}(\mathtt{rent@agt(A)@obj(}y_c\mathtt{)@from(Z)}) \wedge \mathit{Car}(y_c) \wedge \mathit{Blue}(y_c)$$
$$\Phi_2 \equiv \mathit{Achieved}(\mathtt{Z}, (\exists z_c)(\mathit{Sel}(\mathtt{Z}, z_c, \mathtt{w_c}, \mathtt{idA}) \wedge \mathit{Car}(z_c) \wedge \mathit{Blue}(z_c)))$$
$$\Upsilon \equiv (\iota s.\mathit{Sel}(\mathtt{Z}, s, \mathtt{w_c}, \mathtt{idA}) \wedge \mathit{Car}(s) \wedge \mathit{Blue}(s))$$

Zoe's new intention. Zoe's new intention is represented by a DIS—call it D_Z—that, in this case, is identical to the subsidiary box, D_s, seen above. However, it is important to distinguish D_s and D_Z since updates that Zoe makes to her intention will be reflected in D_Z, but need not be reflected in D_s. For example, Zoe might not inform Alice that she plans to facilitate her selection of a car by putting on her glasses. The subsidiary box D_s represents what Alice intends Zoe to achieve for her; this information need not include a complete plan for how Zoe makes her selection. In contrast, D_Z represents Zoe's intention as it evolves over time. The operations $\sigma_{3.3}, \ldots, \sigma_{3.6}$ above can be used to create D_Z, whose "stand-alone" translation into first-order logic is given by:

$$\|D_Z\| = \mathit{IntTh}(\mathtt{Z}, \mathtt{idZ}, (\exists z_c)(\mathit{Sel}(\mathtt{Z}, z_c, \mathtt{w_c}, \mathtt{idA}) \wedge \mathit{Car}(z_c) \wedge \mathit{Blue}(z_c)))$$

Example 4 Alice tells Zoe that the car should be an economy car

Alice. Below are the update operations that are applied to Alice's DIS.[14]

---

13 It would be perfectly fine for the parameter name in Zoe's DIS to be the same as the corresponding parameter name in Alice's DIS. They are shown here as different just to highlight the possibility.

14 The redundancy in these update operations derives from the following: (1) Alice intends that the car be an economy model; (2) Alice intends that Zoe select an economy model; and (3) Alice intends to rent whatever economy model, blue car Zoe selects. It might be possible to remove some of this redundancy by, for example, making the subsidiary box D_s one of the arguments of the *DefVar* specification. However, this would complicate both the syntax and semantics; thus, we do not explore it here.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

[$\sigma_{4.1}$] $AddConds(D_A, \{Econ(\mathtt{w_c})\})$— Add the constraint, $Econ(\mathtt{w_c})$, to D_A.

[$\sigma_{4.2}$] $AddDefVarCond(D_A, \mathtt{w_c}, Econ(\_))$—Add the constraint, $Econ(\_)$, to the *DefVar* entry for $\mathtt{w_c}$.

[$\sigma_{4.3}$] $AddConds(D_s, \{Econ(\mathtt{u_c})\})$—Add the constraint, $Econ(\mathtt{u_c})$, to the subsidiary box, D_s.

Below are the resulting DIS and its translation into first-order logic.

$D_A =$

ID/Agt:	idA / A
DefVars:	$(\mathtt{w_c}, \{Sel(\mathtt{Z}, \_, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), Car(\_), Blue(\_), Econ(\_)\}, \mathtt{Z}, \mathtt{idZ}, \mathtt{u_c})$
ActType:	$\mathtt{rent@obj(w_c)@from(Z)}$
SubBoxes:	*ID/Agt*: idZ / Z; *ExVars*: $\mathtt{u_c}$; *Conds*: $Sel(\mathtt{Z}, \mathtt{u_c}, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA})$, $Car(\mathtt{u_c})$, $Blue(\mathtt{u_c})$, $Econ(\mathtt{u_c})$
Conds:	$Car(\mathtt{w_c}), Blue(\mathtt{w_c}), Econ(\mathtt{w_c})$

$$\|D_A\| = IntTh(\mathtt{A}, \mathtt{idA}, \langle \lambda y_c.\ \Phi'(y_c)\rangle\ \Upsilon')$$

where:

$$\Phi'(y_c) \equiv Done(\mathtt{rent@agt(A)@obj}(y_c)\mathtt{@from(Z)}) \wedge Car(y_c) \wedge Blue(y_c) \wedge Econ(y_c)$$

$$\Upsilon' \equiv (\iota s. Sel(\mathtt{Z}, s, \mathtt{w_c}, \mathtt{idA}) \wedge Car(s) \wedge Blue(s) \wedge Econ(s))$$

Zoe. We assume that Zoe accepts Alice's new constraint that the car be an economy model.[15] Thus, Zoe decides to update her intention accordingly. In this case, the intention update is modeled by the following operation (which is nearly identical to $\sigma_{4.3}$ above).

[$\sigma_{4.4}$] $AddConds(D_Z, \{Econ(\mathtt{u_c})\})$—Add the constraint, $Econ(\mathtt{u_c})$, to D_Z.

Below are the updated version of D_Z and its "stand-alone" translation.

$D_Z =$

ID/Agt:	idZ / Z
ExVars:	$\mathtt{u_c}$
Conds:	$Sel(\mathtt{Z}, \mathtt{u_c}, \text{“}\mathtt{w_c}\text{”}, \mathtt{idA}), Car(\mathtt{u_c}), Blue(\mathtt{u_c}), Econ(\mathtt{u_c})$

$$\|D_Z\| = IntTh(\mathtt{Z}, \mathtt{idZ}, (\exists z_c)(Sel(\mathtt{Z}, z_c, \mathtt{w_c}, \mathtt{idA}) \wedge Car(z_c) \wedge Blue(z_c) \wedge Econ(z_c)))$$

Notice that constraints inserted by the intention-update operations $\sigma_{4.1}, \ldots, \sigma_{4.4}$ are only superficially different. The different versions could be obtained by applying the single predicate abstraction, $\langle \lambda s.\ Econ(s)\rangle$, to the respective terms, $\mathtt{w_c}$, _, $\mathtt{u_c}$, and $\mathtt{u_c}$.

6.2 Main scenario: group travels to Boston

Note. Now that several DIS-creation and DIS-update operations have been demonstrated, the operations for the examples below will only be given English glosses when they are needed for clarification.

Example 1 Alice, Bob and Chris decide to travel to Boston together

Following Grosz and Hunsberger's Coordinated Cultivation of SharedPlans (CCSP) model [13], we assume that in response to such a decision, each of the agents adopts a GAR intention

[15] Again, the focus here is not on the communication between Alice and Zoe, or on any subsequent decisions. Instead, it is on intention-update operations and the resulting intentions, which are represented by DISs.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

that the group does the travel action. The GAR intentions held by the different individuals in the group all have the same content, except for the *Agt* entry. Let g be A, B or C (i.e., Alice, Bob or Chris). Let D^* be the DIS* representing agent g's GAR intention. Then D^* can be generated by the following operations.

$[\tau_{1.1}]$ $D^* = \mathit{NewDIS}^*(\texttt{idGR}, g, \texttt{GR})$
$[\tau_{1.2}]$ $\mathit{AddDefVar}(D^*, \texttt{v}_{\texttt{dn}}, \texttt{Boston})$
$[\tau_{1.3}]$ $\mathit{AddActType}(D^*, \texttt{travel@to}(\texttt{v}_{\texttt{dn}}))$

Below are the resulting D^* and its translation into first-order logic.

$$D^* = \boxed{\begin{array}{ll} \mathit{ID/Agt/Grp}: & \texttt{idGR} / g / \texttt{GR} \\ \mathit{DefVars}: & (\texttt{v}_{\texttt{dn}}, \texttt{Boston}) \\ \mathit{ActType}: & \texttt{travel@to}(\texttt{v}_{\texttt{dn}}) \end{array}}$$

$$\|D^*\| = \mathit{IntTh}^*(g, \texttt{idGR}, \texttt{GR}, \langle \lambda x_{dn}.\, \mathit{Done}(\texttt{travel@agt(GR)@to}(x_{dn}))\rangle\, \texttt{Boston})$$

Example 2 The group decides that they will travel to Boston by getting and driving a car

The group's decision is to travel to Boston by doing two subsidiary actions involving a single car: one of them will get the car (get); and one of them will drive it (drive). This group decision obliges each agent to update its coresponding GAR intention. The composite update can be broken down into the following primitive DIS-update operations:

$[\tau_{2.1}]$ $\mathit{AddExVar}(D^*, \texttt{v}_{\texttt{c}})$, $\mathit{AddExVar}(D^*, \texttt{v}_{\texttt{g}})$, $\mathit{AddExVar}(D^*, \texttt{v}_{\texttt{d}})$—Variables representing the car, the get agent, and the drive agent.
$[\tau_{2.2}]$ $\mathit{AddConds}(D^*, \{\mathit{Car}(\texttt{v}_{\texttt{c}})\})$
$[\tau_{2.3}]$ $D_1 = \mathit{NewDIS}(\texttt{idSub1}, \texttt{v}_{\texttt{g}})$—Subsidiary box for the get action.
$[\tau_{2.4}]$ $\mathit{AddActType}(D_1, \texttt{get@obj}(\texttt{v}_{\texttt{c}}))$
$[\tau_{2.5}]$ $\mathit{AddSubBox}(D^*, D_1)$
$[\tau_{2.6}]$ $D_2 = \mathit{NewDIS}(\texttt{idSub2}, \texttt{v}_{\texttt{d}})$—Subsidiary box for the drive action.
$[\tau_{2.7}]$ $\mathit{AddActType}(D_2, \texttt{drive@obj}(\texttt{v}_{\texttt{c}})\texttt{@to}(\texttt{v}_{\texttt{dn}}))$
$[\tau_{2.8}]$ $\mathit{AddSubBox}(D^*, D_2)$

The resulting DIS* and its translation into first-order logic are given below.

$$D^* = \boxed{\begin{array}{ll} \mathit{ID/Agt/Grp}: & \texttt{idGR} / g / \texttt{GR} \\ \mathit{ExVars}: & \texttt{v}_{\texttt{g}}, \texttt{v}_{\texttt{d}}, \texttt{v}_{\texttt{c}} \\ \mathit{DefVars}: & (\texttt{v}_{\texttt{dn}}, \texttt{Boston}) \\ \mathit{SubBoxes}: & \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \texttt{idSub1} / \texttt{v}_{\texttt{g}} \\ \mathit{ActType}: & \texttt{get@obj}(\texttt{v}_{\texttt{c}}) \end{array}}\,, \\ & \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \texttt{idSub2} / \texttt{v}_{\texttt{d}} \\ \mathit{ActType}: & \texttt{drive@obj}(\texttt{v}_{\texttt{c}})\texttt{@to}(\texttt{v}_{\texttt{dn}}) \end{array}} \\ \mathit{Conds}: & \mathit{Car}(\texttt{v}_{\texttt{c}}) \end{array}}$$

$$\|D^*\| = \mathit{IntTh}^*(g, \texttt{idGR}, \texttt{GR}, \langle \lambda x_{dn}.\, (\exists x_g, x_d, x_c)(\Phi_1 \wedge \Phi_2 \wedge \mathit{Car}(x_c))\rangle \texttt{Boston})$$

where:

$$\Phi_1 \equiv \mathit{Achieved}(x_g, \mathit{Done}(\texttt{get@agt}(x_g)\texttt{@obj}(x_c)))$$

$$\Phi_2 \equiv \mathit{Achieved}(x_d, \mathit{Done}(\texttt{drive@agt}(x_d)\texttt{@obj}(x_c)\texttt{@to}(x_{dn})))$$

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

Translating subsidiary boxes that contain free variables. Notice that in the above example, D^* contains four parameters: $\mathtt{v_g}$, $\mathtt{v_d}$, $\mathtt{v_c}$ and $\mathtt{v_{dn}}$. In the translation of D^* into a logical formula, each occurrence of these variables is translated into an occurrence of x_g, x_d, x_c or x_{dn}, respectively. Notice further that, viewed as stand-alone boxes, the subsidiary boxes within D^* include free-variable occurrences of $\mathtt{v_g}$, $\mathtt{v_d}$, $\mathtt{v_c}$ and $\mathtt{v_{dn}}$, including within the *Agt* and *Grp* fields. Thus, we first extend the syntax rules for these fields:

(2′) The *Agt* field must contain either a single constant, $g \in AgtNames$, or a single variable, $v \in VarNames$.
(3′) The *Grp* field must contain a constant, $GR \in GrpNames$, or a variable, $v \in VarNames$.

Next, we must extend the "in context" translation function to accommodate the presence of free variables within the subsidiary boxes of D^*. The proper "in context" translation of these variables should be consistent with occurrences of those same variables in the parent box, D^*. Thus, occurrences of $\mathtt{v_g}$ inside the subsidiary boxes should be translated into occurrences of x_g, and so forth. To ensure that this happens, we include an *environment* as an optional argument to the "in context" translation function, $\| \cdot \|^c$. An environment is simply a list of pairs, where each pair has the form $(\mathtt{v}, x)$, where $\mathtt{v}$ is a parameter (either *ExVar* or *DefVar*) and x is the logical variable that $\mathtt{v}$ is translated into. When performing the "in context" translation of a subsidiary box, $\mathcal{S}$, that resides within some DIS D, the translation of $\mathcal{S}$ is given by:

> $\|\mathcal{S}, E\|^c \;\equiv\; \|\mathcal{S}\|^c$, except that all free occurrences of parameters in $\mathcal{S}$ must be translated according to the corresponding entry in the environment E—where E contains an entry for each *ExVar* and each *DefVar* parameter in D.

The example given above illustrates the "in context" translation of subsidiary boxes that contain free variables that are "captured" by the parent box.

Example 3 The group decides Alice should be the one to get (and thereby select) the car *updating D^* (i.e., the GAR intentions held by each agent in the group).*

This decision obliges each agent to update its GAR intention to reflect (1) the selection of $\mathtt{A}$ as the value for the agent variable, $\mathtt{v_g}$, in their plan; and (2) that Alice's getting of the car should determine which car they use in their plan (i.e., that the value of the variable $\mathtt{v_c}$ should be whatever car Alice gets). The first update is accomplished by shifting the variable $\mathtt{v_g}$ in the box D^* from the *ExVar* to the *DefVar* category, and giving it the value $\mathtt{A}$ (cf. $\tau_{3.1}$ below). The second update is accomplished by similarly shifting the variable, $\mathtt{v_c}$, from the *ExVar* to the *DefVar* category; however, in this case, the value of $\mathtt{v_c}$ is determined by a set of conditions (cf. $\tau_{3.2}$ below). Simultaneously, the subsidiary box for the `get` action needs to have a new *ExVar* variable representing the car that Alice gets (cf. $\tau_{3.3}$ and $\tau_{3.4}$ below).

[$\tau_{3.1}$] $\mathit{ShiftVar}(D^*, \mathtt{v_g}, \mathtt{A})$
[$\tau_{3.2}$] $\mathit{ShiftVar}(D^*, \mathtt{v_c}, \{\mathit{Done}(\mathtt{get@agt(v_g)@obj(\_)}), \mathit{Car}(\_)\}, \mathtt{A}, \mathtt{idSub1}, \mathtt{w_c})$
[$\tau_{3.3}$] $\mathit{AddExVar}(D_1, \mathtt{w_c})$
[$\tau_{3.4}$] $\mathit{AddConds}(D_1, \{\mathit{Car}(\mathtt{w_c})\})$

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

Below are the updated version of D^* and its translation into first-order logic.

$$D^* = \begin{array}{|ll|}\hline \textit{ID/Agt /Grp}: & \texttt{idGR} \,/\, g \,/\, \texttt{GR} \\ \textit{ExVars}: & \texttt{v}_{\texttt{d}} \\ \textit{DefVars}: & (\texttt{v}_{\texttt{dn}}, \texttt{Boston}), (\texttt{v}_{\texttt{g}}, \texttt{A}), \\ & (\texttt{v}_{\texttt{c}}, \{\textit{Done}(\texttt{get@agt}(\texttt{v}_{\texttt{g}})\texttt{@obj}(\_)), \textit{Car}(\_)\}, \texttt{A}, \texttt{idSub1}, \texttt{w}_{\texttt{c}}) \\ \textit{SubBoxes}: & \begin{array}{|ll|}\hline \textit{ID/Agt}: & \texttt{idSub1} \,/\, \texttt{v}_{\texttt{g}} \\ \textit{ExVars}: & \texttt{w}_{\texttt{c}} \\ \textit{ActType}: & \texttt{get@obj}(\texttt{w}_{\texttt{c}}) \\ \textit{Conds}: & \textit{Car}(\texttt{w}_{\texttt{c}}) \\ \hline \end{array}, \\ & \begin{array}{|ll|}\hline \textit{ID/Agt}: & \texttt{idSub2} \,/\, \texttt{v}_{\texttt{d}} \\ \textit{ExVars}: & \\ \textit{ActType}: & \texttt{drive@obj}(\texttt{v}_{\texttt{c}})\texttt{@to}(\texttt{v}_{\texttt{dn}}) \\ \textit{Conds}: & \\ \hline \end{array} \\ \textit{Conds}: & \textit{Car}(\texttt{v}_{\texttt{c}}) \\ \hline \end{array}$$

$$\|D^*\| = \textit{IntTh}^*(g, \texttt{idGR}, \texttt{GR}, \langle\lambda x_{dn}.\, \langle\lambda x_g.\, \langle\lambda x_c.\, (\exists x_d)(\Phi_1' \wedge \Phi_2 \wedge \textit{Car}(x_c))\rangle\Upsilon\rangle\texttt{A}\rangle\texttt{Boston})$$

where:
$$\begin{array}{lll} \Phi_1' & \equiv & \textit{Achieved}(x_g, (\exists y_c)\textit{Done}(\texttt{get@agt}(x_g)\texttt{@obj}(y_c)) \wedge \textit{Car}(y_c)) \\ \Phi_2 & \equiv & \textit{Achieved}(x_d, \textit{Done}(\texttt{drive@agt}(x_d)\texttt{@obj}(x_c)\texttt{@to}(x_{dn}))) \\ \Upsilon & \equiv & (\imath s.\textit{Done}(\texttt{get@agt}(x_g)\texttt{@obj}(s)) \wedge \textit{Car}(s)) \end{array}$$

Alice. In response to the group decision that Alice should get the car, and insodoing select the car for the group activity, Alice is obliged to get the car. Thus, she adopts a new, subsidiary intention to get a car. Although each of the agents in the group holds the high-level GAR intention, only Alice adopts the subsidiary intention aimed at getting the car.

Let D_1^A be the DIS representing Alice's intention that she get a car. D_1^A is nearly identical to the subsidiary box, D_1, discussed above.[16] The only difference is that the *Agt* field contains the constant A instead of the *DefVar*, $\texttt{v}_{\texttt{g}}$. (Later examples will illustrate cases where there are greater differences between a subsidiary DIS and the corresponding DIS representing an adopted intention.)

$$D_1^A = \begin{array}{|ll|}\hline \textit{ID/Agt}: & \texttt{idSub1} \,/\, \texttt{A} \\ \textit{ExVars}: & \texttt{w}_{\texttt{c}} \\ \textit{ActType}: & \texttt{get@obj}(\texttt{w}_{\texttt{c}}) \\ \textit{Conds}: & \textit{Car}(\texttt{w}_{\texttt{c}}) \\ \hline \end{array}$$

$$\|D_1^A\| = \textit{IntTh}(\texttt{A}, \texttt{idSub1}, (\exists y_c)(\textit{Done}(\texttt{get@agt}(\texttt{A})\texttt{@obj}(y_c)) \wedge \textit{Car}(y_c)))$$

Example 4 Alice decides to get a car by renting it

In response to her decision, not only must Alice update her original intention to reflect that she will be getting a car by renting it, but also she must adopt a subsidiary intention to rent a car that, if satisfied, will necessarily satisfy her original intention. In fact, Alice will normally suspend processing of her intention to get a car, while she focuses on processing her intention to rent a car. The group need not know anything about how Alice is getting a car; thus, this decision of Alice's need not have any impact on the GAR intentions held by each agent in the group.

Updating Alice's intention to get a car. Because Alice intends to get a car by renting it, the variable representing the car in D_1^A must be changed from the *ExVar* category to the *DefVar* category. By doing this, Alice's intention is effectively changed from "I intend to get a

[16] Since the DIS-creation and DIS-update operations are so similar to those used to create D_1, they are not presented here.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

car" to "I intend to get whatever car I rent." Notice that this implicit kind of selection does not employ explicit selection, for example, as represented by the $Sel(\ldots)$ predicate in prior examples. Below are the update, $\tau_{4.1}$, the resulting DIS, and its translation.

$$[\tau_{4.1}]\ \ \mathit{ShiftVar}(D_1^A, \mathtt{w_c}, \{\mathit{Done}(\mathtt{rent@agt(A)@obj(\_)}), \mathit{Car}(\_)\}, \mathtt{A}, \mathtt{idSub1r}, \mathtt{u_c})$$

$$D_1^A = \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \mathtt{idSub1\,/\,A} \\ \mathit{DefVars}: & (\mathtt{w_c}, \{\mathit{Done}(\mathtt{rent@agt(A)@obj(\_)}), \mathit{Car}(\_)\}, \mathtt{A}, \mathtt{idSub1r}, \mathtt{u_c}) \\ \mathit{ActType}: & \mathtt{get@obj(w_c)} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{w_c}) \end{array}}$$

$$\|D_1^A\| = \mathit{IntTh}(\mathtt{A}, \mathtt{idSub1}, \langle \lambda y_c.\, \mathit{Done}(\mathtt{get@agt(A)@obj}(y_c)) \wedge \mathit{Car}(y_c)\rangle \Upsilon_r)$$

where: $\Upsilon_r = (\iota s.\mathit{Done}(\mathtt{rent@agt(A)@obj}(s)) \wedge \mathit{Car}(s))$.

Alice's new intention to rent a car. Let D_{1r}^A be the new DIS representing Alice's new intention to rent a car. Like the DIS for her previous intention to get a car, D_{1r}^A will contain an existential variable representing the car, a constraint that the item be a car, and an act type—in this case, representing her rental action. Below are D_{1r}^A and its translation into first-order logic.

$$D_{1r}^A = \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \mathtt{idSub1r\,/\,A} \\ \mathit{ExVars}: & \mathtt{u_c} \\ \mathit{ActType}: & \mathtt{rent@obj(u_c)} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{u_c}) \end{array}}$$

$$\|D_{1r}^A\| = \mathit{IntTh}(\mathtt{A}, \mathtt{idSub1r}, (\exists z_c)(\mathit{Done}(\mathtt{rent@agt(A)@obj}(z_c)) \wedge \mathit{Car}(z_c)))$$

Recalling the side scenario involving Alice and Zoe. A quick glance back at Alice's intention to rent a car that began the side scenario from Sect. 6.1 will reveal that the DIS in that section is essentially the same as D_{1r}^A, above—the only difference being the choice of names for the constants. Thus, the main scenario now continues, assuming that Alice has decided to rent a car by having Zoe select it (a blue car, of course); and that Zoe adopts an intention to select such a car for Alice; and that Alice later informs Zoe that she wants the car to be an economy car.

Example 5 The group decides that Bob should drive the car

As with the group's decision that Alice should get the car, their decision that Bob should drive the car obliges each agent to update its corresponding GAR intention. In addition, it obliges Bob to adopt a new, subsidiary intention aimed at the "drive" action. The updating of the GAR intention held by each agent is similar to the updating in the group's selection of Alice to do the "get" action; however, the representation of Bob's intention concerning the "drive" action is more complex, primarily because Bob is supposed to drive to whatever destination is chosen by the group, using whatever car is chosen by whatever agent is chosen by the group to do the "get" action. Notice that Alice, the car and the destination are all represented by free variables in the subsidiary box corresponding to the "drive" action (in the most recent version of D^*). As of this moment, the group has chosen Alice to do the "get" action, but, let us suppose, she has not yet chosen a car. Similarly, the group has chosen Boston as its destination. The representation of Bob's intention should not simply hardwire the current choices for the values of the relevant free variables since the group might later decide to change those values. For example, the group might decide to select a different agent for the "get" action, thereby invalidating Alice's current choice of a car; or the group might decide to go to New York, thereby invalidating the current choice of destination. Similarly, if Alice had already chosen a car, Bob's intention must not simply hardwire that choice of

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

car since she might later change her mind. Thus, the DIS representing Bob's intention must be carefully constructed to be robust to these sorts of changes.

Updating the GAR intentions. The updating of the GAR intentions in this case is much simpler than in the earlier case in which the group selected Alice to do the "get" action (cf. Example 3, above) since the group is not requiring Bob to select values for any parameters. The only required update is to shift the parameter, $\mathtt{v_d}$, from the *ExVar* to the *DefVar* category.

$$[\tau_{5.1}] \;\; \mathit{ShiftVar}(D^*, \mathtt{v_d}, \mathtt{B})$$

Here's the updated version of D^*, representing the GAR intention held by each agent in the group, together with its translation into first-order logic.

$$D^* = \begin{array}{|ll|}\hline \mathit{ID/Agt/Grp}: & \mathtt{idGR} \,/\, g \,/\, \mathtt{GR} \\ \mathit{DefVars}: & (\mathtt{v_{dn}}, \mathtt{Boston}), (\mathtt{v_g}, \mathtt{A}), \\ & (\mathtt{v_c}, \{\mathit{Done}(\mathtt{get@agt(v_g)@obj(\_)}), \mathit{Car}(\_)\}, \mathtt{A}, \mathtt{idSub1}, \mathtt{w_c}), \\ & (\mathtt{v_d}, \mathtt{B}) \\ \mathit{SubBoxes}: & \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{idSub1} \,/\, \mathtt{v_g} \\ \mathit{ExVars}: & \mathtt{w_c} \\ \mathit{ActType}: & \mathtt{get@obj(w_c)} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{w_c}) \\ \hline \end{array}\;, \\ & \begin{array}{|ll|}\hline \mathit{ID/Agt}: & \mathtt{idSub2} \,/\, \mathtt{v_d} \\ \mathit{ExVars}: & \\ \mathit{ActType}: & \mathtt{drive@obj(v_c)@to(v_{dn})} \\ \mathit{Conds}: & \\ \hline \end{array} \\ \mathit{Conds}: & \mathit{Car}(\mathtt{v_c}) \\ \hline \end{array}$$

$$\|D^*\| = \mathit{IntTh}^*(g, \mathtt{idGR}, \mathtt{GR}, \langle \lambda x_{dn}.\, \langle \lambda x_g.\, \langle \lambda x_c.\, \langle \lambda x_d.\, \Phi_1' \wedge \Phi_2 \wedge \mathit{Car}(x_c) \rangle \mathtt{B} \rangle \Upsilon \rangle \mathtt{A} \rangle \mathtt{Boston})$$

where:

$$\begin{aligned} \Phi_1' &\equiv \mathit{Achieved}(x_g, (\exists y_c)\mathit{Done}(\mathtt{get@agt}(x_g)\mathtt{@obj}(y_c)) \wedge \mathit{Car}(y_c)) \\ \Phi_2 &\equiv \mathit{Achieved}(x_d, \mathit{Done}(\mathtt{drive@agt}(x_d)\mathtt{@obj}(x_c)\mathtt{@to}(x_{dn}))) \\ \Upsilon &\equiv (\imath s.\mathit{Done}(\mathtt{get@agt}(x_g)\mathtt{@obj}(s)) \wedge \mathit{Car}(s)) \end{aligned}$$

Bob's new intention concerning the "drive" action. As it is currently configured, the box D_2, defined as the subsidiary box in D^* corresponding to the "drive" action, cannot serve as the stand-alone DIS representing Bob's intention to drive a car. The problem is that D_2 contains free variables—in particular, $\mathtt{v_d}$, $\mathtt{v_c}$ and $\mathtt{v_{dn}}$. Although the "in context" translation of D_2 as a subsidiary box within D^* leads to the correct $\mathit{Achieved}(\ldots)$ clause within the GAR intention represented by D^*, the generation of an intention clause requires using the "stand-alone" translation. And the "stand-alone" translation function is not (and should not be) defined for a DIS containing free variables. Thus, a new DIS must be generated that is similar to D_2, but in which the free variables have been converted to something else. After all, Bob's intention is to drive not any old car, but whatever car Alice has chosen for the group; and Bob's intention is to drive that car to the destination chosen by the group. Furthermore, since the proper specification of the value of the car refers to the group's choice for the agent doing the "get" action, which is represented by the parameter, $\mathtt{v_g}$, in D^*, that parameter too must be properly defined in the DIS for Bob's intention.

The robust solution is to start with a copy of the subsidiary box D_2 and convert the free variables, $\mathtt{v_{dn}}$, $\mathtt{v_g}$ and $\mathtt{v_c}$, into *DefVar* parameters whose values are drawn from the corresponding *DefVar* entries in D^*. (Since Bob is one of the agents holding the GAR intention represented by D^*, it is proper to assume that the information contained in D^* is available to Bob.) However, the free variable, $\mathtt{v_d}$, which represents the agent of the "drive" action, can simply be hardwired as $\mathtt{B}$, since any change in the value of that parameter would lead Bob to drop his intention anyway.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

First, we introduce the following new syntax for a *DefVar* entry:

(5c) $(v, \mathit{defVarValue}(id, v))$

The intended interpretation of such an entry, which will be formalized in the extended translation function, is that the value of the parameter v shall be whatever value v has in the DIS identified by id.

Next, let D_2^B be a copy of the subsidiary DIS D_2 in which the *Agt* field has been altered to contain the constant, B (i.e., Bob). Here are the DIS-update operations that "capture" the free variables, $\mathtt{v_{dn}}$, $\mathtt{v_g}$ and $\mathtt{v_c}$.

[$\tau_{6.1}$] $\mathit{AddDefVar}(D_2^B, \mathtt{v_{dn}}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_{dn}}))$
—Add a *DefVar* entry for $\mathtt{v_{dn}}$, thus capturing any of its formerly free occurrences in D_2^B. Specify its value to be whatever value is being used in the parent DIS for the variable of the same name.
[$\tau_{6.2}$] $\mathit{AddDefVar}(D_2^B, \mathtt{v_g}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_g}))$—Ditto for $\mathtt{v_g}$.
[$\tau_{6.3}$] $\mathit{AddDefVar}(D_2^B, \mathtt{v_c}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_c}))$—Ditto for $\mathtt{v_c}$.

Each of these operations lead to a *DefVar* entry in D_2^B containing the new *defVarValue* syntax. The translation of this new kind of syntax will be addressed momentarily.

Here is the updated version of D_2^B together with its translation.

$$D_2^B = \boxed{\begin{array}{ll} \mathit{ID/Agt}: & \mathtt{idSub2} / \mathtt{B} \\ \mathit{DefVars}: & (\mathtt{v_{dn}}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_{dn}})) \\ & (\mathtt{v_g}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_g})) \\ & (\mathtt{v_c}, \mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_c})) \\ \mathit{ActType}: & \mathtt{drive@obj(v_c)@to(v_{dn})} \end{array}}$$

$$\|D_2^B\| \equiv \mathit{IntTh}(\mathtt{B}, \mathtt{idSub2}, \langle \lambda y_{dn}.\ \langle \lambda y_g.\ \langle \lambda y_c.\ \Phi_2'' \rangle \Upsilon' \rangle \mathtt{A} \rangle \mathtt{Boston})$$

$$\text{where:} \quad \begin{array}{ll} \Phi_2'' & \equiv \mathit{Done}(\mathtt{drive@agt(B)@obj(}y_c\mathtt{)@to(}y_{dn}\mathtt{)}) \\ \Upsilon' & \equiv (\imath s.\mathit{Done}(\mathtt{get@agt(}y_g\mathtt{)@obj(}s\mathtt{)}) \wedge \mathit{Car}(s)) \end{array}$$

The translation of defVarValue(...) *expressions.* The translation of *defVarValue*(...) expressions is analogous to the evaluation (and *expansion*) of *macros* in the Lisp programming language [34]. For example consider the expression, $\mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_g})$, in D_2^B. The first step in the translation of this expression is to replace it by the corresponding expression from the *DefVar* entry for the parameter, $\mathtt{v_g}$, in the DIS, D^*, identified by idGR. That *DefVar* entry is $(\mathtt{v_g}, \mathtt{A})$. Thus, the "macro expansion" of $\mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_g})$ is simply A. That expression is then translated as usual by the "stand-alone" translation function, yielding A.

Next, consider the expression, $\mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_c})$, in D_2^B. Ignoring the optional arugments, the *DefVar* entry for the variable named $\mathtt{v_c}$ in D^* is: $(\mathtt{v_c}, \{\mathit{Done}(\mathtt{get@agt(v_g)@obj(\_)}), \mathit{Car}(\_)\}$. Thus, the "macro expansion" of the expression, $\mathit{defVarValue}(\mathtt{idGR}, \mathtt{v_c})$, is simply, $\{\mathit{Done}(\mathtt{get@agt(v_g)@obj(\_)}), \mathit{Car}(\_)\}$. This set of conditions then gets translated by the "stand-alone" translation function, as usual, yielding:

$$(\imath s.\mathit{Done}(\mathtt{get@agt(}y_g\mathtt{)@obj(}s\mathtt{)}) \wedge \mathit{Car}(s))$$

Notice that the logical variable into which $\mathtt{v_g}$ is translated is determined by this application of the translation function to D_2^B, which is independent of how $\mathtt{v_g}$ gets translated in D^*.

Notice that should the group subsequently decide to change any of their decisions (e.g., who should do the "get" action; or where they should travel to), these changes would automatically be reflected in Bob's intention, without requiring any changes to his DIS. That is, the translation of Bob's DIS would automatically reflect the changes because the *defVarValue* expressions would generate different terms.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

7 Intentions in situ

In this article we have focused primarily on the partial and dynamic nature of individual intentions without concern about how any particular intention might interact with other intentions. We believe that an adequate solution to the problems of partiality and dynamics is somewhat orthogonal to the solution of other important problems, such as the proper axiomatization of intention or the revision of intentions. In a companion paper [27], we explore these other issues more deeply, in the context of DIS theory. In this section we present a brief overview of the axiomatization of intention and the process of intention revision covered in the companion paper.[17]

Remark on how DISs are used by an agent. We imagine that agents, during the elaboration and negotiation of shared plans, manipulate their respective DISs using the given update operations to modify the content of their intentions. At some point (e.g., as indicated by the particular agent's architecture) agents may wish to consider the consequences of new, individual intentions in the context of their other intentions or beliefs. To consider the logical consequences of a DIS, an agent need only translate the DIS, according to the semantics we have given, into first-order logic.[18] To revise an existing collection of intentions, a process of intention revision, as summarized below and presented in our companion paper [27], is used to compute the maximal subset(s) of existing intentions consistent with the new intention.

7.1 Axiomatization

As we have indicated, intentions in our work have the general form, $\mathit{IntTh}(G, \phi)$, where G is an agent, ϕ is a formula, and *IntTh* is a modal operator. In this section we dispense with reference to the intention identifier; this is always possible, as discussed in Sect. 2.2. The truth of a formula, ϕ—where ϕ can contain modal operators—is then, as usual, expressed relative to a model, M, and a possible world, w, taken from a set of possible worlds, W: $M, w, \models \phi$. The semantics for intention formulas is expressed by way of an intention *accessibility* relation, $\mathcal{I}_G \subseteq W \times W$, such that (we suppress reference to a model) $w \models \mathit{IntTh}(G, \phi)$ just in case $w' \models \phi$ in all worlds w' such that $\mathcal{I}_G(w, w')$.

We adopt the weakest normal modal logic, known as System K, subscribing only to the following axiom of consequential closure [3]:

$$(\mathit{IntTh}(G, \phi) \wedge \mathit{IntTh}(G, \phi \Rightarrow \psi)) \;\Rightarrow\; \mathit{IntTh}(G, \psi)$$

That is, if an agent intends that ϕ, and also intends that $\phi \Rightarrow \psi$, then the agent must also intend that ψ. An example might be the following, "If I intend that the house be clean by 2:00 p.m., and I also intend that if the house is clean by 2:00 p.m. then I will go to the store at 2:00 p.m., then I also intend that I will go to the store at 2:00 p.m." A number of other typical axioms of modal logic all appear too strong and hence are not adopted here. For example, we do not adopt the axiom, $\models \mathit{IntTh}(G, \phi) \Rightarrow \phi$, since a rational agent will not adopt an intention that ϕ if ϕ is already true.

Rather than remaining within a modal logic, we instead adopt a reified approach to possible worlds; in this way both the semantics of DISs and inference remains within first-

[17] Our treatment in this article is restricted to DISs which do not involve combinations of logical operators. The companion paper [27] relaxes that restriction by augmenting the syntax to allow implication and negation over DISs. Thus, if Φ and Ψ are DISs, then so are $\neg\Phi$ and $\Phi \rightarrow \Psi$. Such DISs are then translated into FOL formulas involving the $\neg$ and $\Rightarrow$ operators.

[18] There are, of course, alternatives to this approach. One would involve developing a proof theory for DISs analagous to that developed for DRTs [37].

order logic. We adopt a reified modal logic [24,26] in which ϕ becomes a function and *IntTh* is introduced as a new predicate. The process of reification translates a statement, $w \models IntTh(G, \phi)$, into a FOL formula, $IntTh(G, \Phi, w)$, where Φ is a function which can be interpreted as a set of possible worlds—intuitively, the set of possible worlds where ϕ holds—and $IntTh(G, \Phi, w) \equiv (\forall w')acc(G, w, w')$, where $acc(G, w, w')$ is now a formula which is true just in case w' is accessible to G from w. For simplicity, we also adopt a common names assumption for each possible world [3]. This has the consequence that agents share cross-world identification of objects in the universe of discourse. Our approach differs significantly from Kamp's mental structures approach for representing modalities in DRT [19]: our approach is closer in spirit to standard methods adopted in the AI knowledge representation literature; furthermore, we address intention revision, whereas Kamp does not.

Finally, note that time is not expressed explicitly in the logic, other than through the assumption that an agent's set of intentions/DISs correspond to those that are true *now;* that set is assumed to persist until otherwise modified through an update or revision operation. It would be straightforward to extend DIS theory to represent time explicitly: see, for example, approaches within a reified logic [26].

7.2 Intention revision

We adopt a syntactic form of intention revision modeled on similar proposals from the belief revision literature [11,25,26]. Most approaches to belief revision are founded on the idea of *minimal change*: to revise a set of beliefs, S, with some new proposition, p, where p is inconsistent with S, one should make the minimal change necessary to S to accommodate p. In syntactic belief revision (also called base revision) the syntactic form of an agent's beliefs (or intentions in our case) is important and is preserved. Briefly, the idea is that if a belief base contains $\{p, p \Rightarrow q\}$ and p is removed, then q will also, as it loses its "support" from p. In effect, one starts with a new set of beliefs containing just $\neg p$ and then iteratively adds contents from S, checking consistency along the way. In contrast, a model-based or belief set revision approach compares models in terms of the minimal changes that need to be made to accommodate the new belief. In this case, we have the initial model, $\{p, q\}$, which can be minimally modified to $\{\neg p, q\}$. In both approaches, the initial set of beliefs can be ordered according to some preference. For example, there might be certain causal and inviolable rules that an agent would never wish to disregard; those would be given highest priority.

Intention revision takes place in two steps within our framework. Let S be the current set of an agent's intentions, in DIS form. Suppose we wish to modify an existing DIS, $D \in S$, according to one of the update functions described in Sect. 5. Let D' correspond to the updated DIS. To revise S, we create a set of equivalence classes on $S: \{S_1, S_2, \ldots, S_n\}$ such that S_1 is meant to correspond to those elements of S that are most important and S_n those that are least important. We start with D' and augment it with the maximal subset of S_1 such that the result is consistent and where consistency is determined via the translation of DISs to FOL. We then repeat the process for each maximal subset of the next equivalence class and stop when no additional elements of S can be added without introducing an inconsistency. The resulting set corresponds to the output of the revision process, though it may not be unique.

The ordering relation on S which induces the equivalence classes is a preorder chosen in roughly the following way. The set S_1 consists of all implications or rules involving intentions and the next levels reflect the level of decomposition in any intention. For example, if I intend to travel to Boston by driving a car then my top level intention of traveling to Boston takes precedence over my intention to do so by driving. Only one modification of the standard DIS form is needed to provide a sufficiently fine-grain ordering. In particular, we rewrite DISs in

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

terms of binary relations so that each element of a DIS (e.g., the list of agents) is stored as a binary relation (e.g., agents(`DIS25,Zoe`)). The resulting ordering satisfies the conditions described by Nebel [25] and implies that the AGM properties [9] of the resulting revision operation are respected.

Our motivation for adopting a syntactic form of intention revision is that we believe it provides an attractive approach to solving the well known intention "side-effects" problem. A typical example is given by Cohen and Levesque [4]: a person intends to go to the dentist and also knows that the visit will be painful. However, the person does not also intend the side-effect of receiving pain: if he did then if he should decide not to go to the dentist, he would be compelled to explore other ways to receive pain (since intentions persist)! In contrast, in base revision the initial contents of the agent's intentions and beliefs would include a rule of the form, "if I intend to α then I believe, *ceteris paribus*, that I will obtain the consequences of performing α (i.e., pain)." If the intention is subsequently retracted, then so too must the belief indicated in the consequent.[19]

8 Conclusions

A long line of research on the representation of intentions [4,7,22,30,32,33,36], starting with the seminal work of Cohen and Levesque [4], has centered on the important property of persistence of intention and also on the role of intentions in the deliberations of a rational agent.

More recently, de Boer et al. [6] presented an approach to modeling interactions in multi-agent systems based on process algebra and constraint programming. Their work focuses on synchronized communication and action execution in two distinct phases. The first phase is a negotiation phase in which agents independently propose sets of constraints to impose on the parameters of a single joint action. If the constraints imposed by all of the agents are consistent, the agents move to an execution phase. In the execution phase, each agent independently and simultaneously chooses a set of values for some or all of the action parameters. If the choices of all the agents in this phase are unifiable and satisfy the above-mentioned constraints, then the joint action succeeds; otherwise it fails. Certain elements of their work bear some similarity to the model presented in this article. For example, if an agent leaves a parameter free during the execution phase, then the value for that parameter might be determined by another agent. In addition, the constraints proposed by agents are similar to the propositions in the *Conds* field of a DIS. However, there are many more differences. For example, our work accommodates the interleaving of planning and execution, explicitly models the delegation of authority and responsibility for binding parameters, explicitly represents intentions and intention-creation and intention-update operations, and accommodates hierarchical action decomposition involving tasks done by different agents.

Our work takes as its starting point the observation that people elaborate and revise their intentions in an *incremental* fashion: intentions will often be only partially specified, requiring the use of existential quantifiers within the scope of an intention; however, during intention elaboration, subsequent references to the same intention will require access to the elements within the scope of such quantifers. This article examined various examples that illustrated these properties, beginning with such simple statements as, "Alice intends to drive the car that Bob picks", in which the referenced object has not yet been identified. We also observed that

[19] As we discuss in the companion paper [27], a proper treatment of this example requires a representation, along the lines of DISs, corresponding to belief.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

such statements share some features of well known examples from the linguistics literature and the various dynamic logics, notably Kamp's DRT, developed to handle them [12,18,20]. We developed DISs to provide a similar flexibility of representation and elaboration involving agent intentions.

A further focus of our work originates from the—reasonable we think—observation that agents working in a team will often delegate the choice of particular object or property of an object referred to by an intention to other agents; any adequate theory of intention must provide mechanisms to support the consistent reference to such objects across groups during the multi-agent elaboration process. DISs provide a framework that enforces such dependencies during elaborations and execution.

In a companion paper [27], we show how these properties of DISs lend themselves nicely to a treatment of intention revision and intention side-effects that takes as its starting point the observation that the elements that constitute the content of an intention are "not all created equal." In that paper we also extend DISs to support the representation of arbitrary logical combinations (within and outside the scope of the intention operator) of intentions and their logical consequences.

We believe a secondary contribution of this work is that it brings together, for the first time, several independent threads of research. The concept of intentional context plays a prominent role in certain theories of collaboration, notably the theory of SharedPlans [13–15,17], as well as in its application to discourse understanding and collaborative interface design [1,29,31]. In contrast, the notion of attentional state introduced in the work of Grosz and others [16,23] to reflect the salience of objects in a natural language discourse has never played a first-class role in the theory of SharedPlans. In addition, the contributions of Kamp [20] and Heim [18] in linguistics have both been developed independently of the above contributions from the fields of multiagent systems and computational linguistics. We believe that the DIS theory that we have presented represents a first attempt at bringing together, in a productive fashion, these related but independently developed and motivated theories.

References

1. Babaian, T., Grosz, B. J., & Shieber, S. (2002). A writer's collaborative assistant. In *Proceedings of the international conference on intelligent user interfaces, IUI'02*. ACM Press.
2. Bratman, M. E. (1999). *Faces of intention: Selected essays on intention and agency*. Cambridge: Cambridge University Press.
3. Chellas, B. F. (1980). *Modal logic: An introduction*. Cambridge: Cambridge University Press.
4. Cohen, P. R., & Levesque, H. J. (1990). Intention is choice with commitment. *Artificial Intelligence, 42*, 213–261.
5. Cohen, P. R., & Levesque, H. J. (1991). Teamwork. *Nous, 25*, 487–512.
6. de Boer, F. S., de Vries, W., Meyer, J.-J. Ch., van Eijk, R. M., & van der Hoek, W. (2005). Process algebra and constraint programming for modeling interactions in mas. *Applicable Algebra in Engineering, Communication and Computing, 16*(2–3), 113–150.
7. Dignum, F., Meyer, J.-J. Ch., Wieringa, R. J., & Kuiper, R. (1996). A modal approach to intentions, commitments and obligations: Intention plus commitment yields obligation. In M. A. Brown & J. Carmo (Eds.), *Deontic logic, agency and normative systems* (pp. 80–97). Springer-Verlag.
8. Fitting, M., & Mendelsohn, R. L. (1998). *First-order modal logic*, Vol. 277 of *Synthese Library: Studies in epistemology, logic, methodology, and philosophy of science*. Dordrecht: Kluwer.
9. Gärdenfors, P. (Ed.) (1992). *Belief revision*. Cambridge: Cambridge University Press.
10. Gilbert, M. (2000). *Sociality and responsibility*. New York: Rowman & Littlefield Publishers, Inc.
11. Ginsberg, M. L. (1986). Counterfactuals. *Artificial Intelligence, 30*, 35–79.
12. Groenendijk, J., & Stokhof, M. (1991). Dynamic predicate logic. *Linguistics and Philosophy, 14*(1), 39–100.

Journal: AGNT MS: 9032 CMS: 10458_2008_9032_Article TYPESET DISK LE CP Disp.:2008/2/14 Pages: 29 Layout: Small

13. Grosz, B. J., & Hunsberger, L. (2006). The dynamics of intention in collaborative activity. *Cognitive Systems Research, 7*(2–3), 259–272.
14. Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence, 86*, 269–357.
15. Grosz, B. J., & Kraus, S. (1999). The evolution of Shared Plans. In M. Wooldridge & A. Rao (Eds.), *Foundations of rational agency*, Number 14 in Applied Logic Series, (pp. 227–262). Dordrecht: Kluwer.
16. Grosz, B. J., & Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics, 12*, 175–204.
17. Grosz, B. J., & Sidner, C. L. (1990). Plans for discourse. In P. R. Cohen, J. Morgan, & M. E. Pollack (Eds.), *Intentions in Communication*. Cambridge, MA: The MIT Press.
18. Heim, I. (1983). File change semantics and the familiarity theory of definiteness. In R. Baurle, C. Schwarze, & A. von Stechow (Eds.), *Meaning, use and the interpretation of language* (pp. 164–189). Berlin: de Gruyter.
19. Kamp, H. (1990). In *Prolegomena to a structural account of belief and other attitudes* (Ch. 2, pp. 27–90). Number 20 in CSLI Lecture Notes. Chicago: University of Chicago Press.
20. Kamp, H., & Reyle, U. (1993). *From discourse to logic*, Vol. 42 of *Studies in Linguistics and Philosophy*. Dordrecht, The Netherlands: Kluwer Academic Publishers.
21. Kinny, D., Ljungberg, M., Rao, A. S., Sonenberg, E., Tidhar, G., & Werner, E. (1994). Planned team activity. In C. Castelfranchi & E. Werner (Eds.), *Artificial social systems*, Vol. 830 of *Lecture Notes in Artificial Intelligence*. Amsterdam: Springer Verlag.
22. Konolige, K., & Pollack, M. E. (1993). A representationalist theory of intentions. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI-93)* (pp. 390–395). San Mateo: Morgan Kaufmann.
23. Lochbaum, K. E. (1998). A collaborative planning model of discourse structure. *Computational Linguistics, 24*(4), 525–572.
24. Moore, R. C. (1985). A formal theory of knowledge and action. In *Formal Theories of the Commonsense World*. Norwood: Ablex Publishing Corporation.
25. Nebel, B. (1992). In P. Gärdenfors (Ed.), *Syntax based approaches to belief revision* (pp. 52–88).
26. Ortiz, C. L. (1999). Explanatory update theory: Applications of counterfactual reasoning to causation. *Artificial Intelligence, 108*, 125–178.
27. Ortiz, C. L., & Hunsberger, L. Dynamic intention structures II: A theory of intention revision. In preparation.
28. Ortiz, C. L., Jr. (1999). Introspective and elaborative processes in rational agents. *Journal of the Annals of Mathematics and Artificial Intelligence*.
29. Ortiz, C. L., Jr., & Grosz, B. J. (2000). Interpreting information requests in context: A collaborative web interface for distance learning. *Autonomous Agents and Multi-agent Systems*.
30. Rao, A. S., & Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In J. F. Allen, R. Fikes, & E. Sandewall, (Eds.), *Proceedings of the 2nd International Conference on Knowledge Representation and Reasoning (KR-91)* (pp. 473–484). Los Altos: Morgan Kaufmann Publishers.
31. Rich, C., Sidner, C. L., & Lesh, N. (2001). COLLAGEN: Applying collaborative discourse theory to human-computer interaction. *AI Magazine, 22*(4), 15–26.
32. Sadek, M. D. (1992). A study in the logic of intentions. In *Proceedings of the 3rd Conference on Principles of Knowledge Representation and Reasoning (KR'92)*. Los Altos: Morgan Kaufman Publishers, Inc.
33. Singh, M. P. (1994). *Multiagent systems: A theoretical framework for intentions, know-how, and communications*. Berlin: Springer-Verlag.
34. Steele, G. L. (1990). *Common Lisp the Language*. Belford: Digital Press.
35. Tuomela, R. (1995). *The importance of Us: A philosophical study of basic social notions*. Stanford: Stanford University Press.
36. van der Hoek, W., Jamroga, W., & Wooldridge, M. (2007). Towards a theory of intention revision. *Synthese, 155*(2), 265–290.
37. van Eijck, J. (2005). Discourse representation theory. In K. Brown (Ed.), *Encyclopedia of Language and Linguistics*. Amsterdam: Elsevier. (2 edn.).