The logic of tasks

Giorgi Japaridze*

Department of Computing Sciences, Villanova University
800 Lancaster Avenue, Villanova, PA 19085, USA
http://www.csc.vill.edu/faculty/japaridz/html/home.html

Abstract

The paper introduces a semantics for the language of classical first order logic supplemented with the additional operators $\sqcap$ and $\bigsqcap$. This semantics understands formulas as *tasks*. An agent (say, a machine or a robot), working as a slave for its master (say, the user or the environment), can carry out the task $\alpha\sqcap\beta$ if it can carry out any one of the two tasks α, β, depending on which of them was requested by the master; similarly, it can carry out $\bigsqcap x\alpha(x)$ if it can carry out $\alpha(x)$ for any particular value for x selected by the master; an agent can carry out $\alpha\rightarrow\beta$ if it can carry out β as long as it has, as a slave (resource), an agent who carries out α; finally, carrying out P, where P is an atomic formula, simply means making P true; in particular, $\bot$ is a task that no agent can carry out. When restricted to the language of classical logic, the meaning of formulas is isomorphic to their classical meaning, which makes our semantics a conservative extension of classical semantics.

This semantics can claim to be a formalization of the resource philosophy associated with linear logic, if resources are understood as agents carrying out tasks. The classical operators of our language correspond to the multiplicative operators of linear logic, while $\sqcap$ and $\bigsqcap$ correspond to the additive conjunction and universal quantifier, respectively.

Our formalism may also have a potential to be used in AI as an alternative logic of planning and action. Its main appeal is that it is immune to the frame problem and the knowledge preconditions problem.

The paper axiomatically defines a logic **L** in the above language and proves its soundness and completeness with respect to the task semantics in the following intuitive sense: $\mathbf{L} \vdash \alpha$ iff α can be carried out by an agent who has nothing but its intelligence (i.e. no physical resources or external sources of information) for carrying out tasks. This logic is shown to be semidecidable in the full language and decidable when the classical quantifier (but not $\bigsqcap$) is forbidden in it.

*Supported by Summer Research Grant from Villanova University

1 Introduction

Classical logic is the logic of *facts*, expressed in natural languages with narrative sentences: "The house is clean", "The monster is dead", etc.

The logic I am going to introduce is the logic of *tasks*, normally expressed in natural languages with imperative sentences: "Clean the house!", "Kill the monster!", etc.

While facts have the values *true* or *false*, tasks have the values *accomplished* or *failed*.

Even though natural languages may be using different grammatical forms for expressing facts and tasks, such as the narrative and imperative forms, the formal language we employ does not make such a distinction, and every atomic fact α is, at the same time, considered the atomic task that is accomplished if and only if α is true.

The difference between classical logic and our logic is in the underlying philosophy, which is deterministic for the former and nondeterministic for the latter. In classical logic, the *true/false* value of every sentence is predetermined, while in the logic of tasks the initial semantical value of a sentence may be neither *accomplished* nor *failed*, and become one of them only later. For example, the value of the task "Become a millionaire" for me at this point is not determined, — otherwise any activities directed towards accomplishing this task would be meaningless; whether this task is accomplished or not can be judged only in the eventual situation — say, when I die or when the world ends. Facts can be considered a special sort of tasks whose values are predetermined. This nondeterministic philosophy and distinction between facts and (proper) tasks, however, has no reflection in the formal semantics we choose: the value of a task is only assessed in the eventual situation, so that every task is then either accomplished or failed, and the semantics does not care whether this value was determined "from the very beginning" or became so "later".

Using words such as "initially" or "later" might have suggested that the logic we are talking about is a sort of temporal logic, which, however, is not the case. Our approach, just as the approach of classical logic, has no concept of time: the *accomplished/failed* values, just as the *true/false* values in classical logic, are not relative to *when*. If some sentences of the natural language, such as "The house is clean", can be true today but false tomorrow, that only means that they are incomplete and hence meaningless. The way to make such an expression meaningful is to either assume a fixed time context, or to explicitly specify the latter, as in the sentence "The house is clean at the noon of March 1, 2000", whose value is no longer time relative. Once such a sentence becomes true (accomplished), it stays so forever; and it is false (failed), if it has never become true. Intuitively this means that tasks expressed with imperative sentences such as "Become a millionaire" or "Clean the house" should be considered accomplished if and only if their narrative counterparts, such as "I have become a millionaire" or "The house is (has been) clean(ed)" are true in

a certain situation that we call the *eventual situation*, and it does not matter when or by whom they were made true.

Thus, at the level of atomic facts/tasks, we have a perfect isomorphism between classical formal semantics and our semantics, where the values *true* and *false* correspond to the values *accomplished* and *failed*, respectively.

All the operators of classical logic are also used in our language, and the above isomorphism extends to compound expressions built from those operators, too:

- $\bot$ is a task that is never accomplished;
- the task $\alpha \wedge \beta$ is accomplished iff both α and β are accomplished;
- the task $\alpha \rightarrow \beta$ is accomplished iff β is accomplished whenever α is accomplished;
- the task $\forall x \alpha(x)$ is accomplished iff the task $\alpha(x)$ is accomplished for all particular values of x, etc.

Of course merely replacing "true" by "accomplished" would hardly take us to any place new and interesting. The novelty starts when we extend the language with certain special operators that have no classical counterparts.

The main operator of this kind is $\sqcap$, resembling by its shape and, in a certain sense, also by meaning, the operator $\wedge$.

Notice that when we talk about tasks, we have two agents in mind: the agent who has to accomplish the task (clean the house), and the agent who sets the task (requests "Clean the house!"). The former can be called the *slave*, and the latter the *master*. A good philosophical point of departure is to think of the slave as a machine or a robot, and think of the master as the user or the environment. At the level of atomic tasks, the presence of the two agents is not really visible: as we don't care about timing, the request "Clean the house!" is assumed to be automatically made, so that the master hardly plays any role.

As for

$$\alpha \sqcap \beta,$$

this is a task that signifies no particular request for the slave, but rather two potential requests that the master can make, i.e. two potential tasks that the slave may have to carry out: α and β. The slave is able to carry out this task, if it can accomplish any particular one of the two tasks α and β, depending on which of them is requested by the master. This task is generally easier to carry out than $\alpha \wedge \beta$. The latter obligates the slave to accomplish both of the tasks α and β (two tasks), while the former only obligates it to accomplish either of the tasks α and β (one task). For example, the agent may have enough time and energy to clean either the house or the lawn, — any one of the two, — but it

may not have sufficient resources to clean both the house and the lawn. In this case it can carry out

$$\textit{Clean the house} \sqcap \textit{Clean the lawn},$$

but cannot accomplish

$$\textit{Clean the house} \wedge \textit{Clean the lawn}.$$

Moreover, this agent can carry out the task

$$\textit{Clean the house} \sqcap \neg \textit{Clean the house},$$

while no agent can ever accomplish the contradictory task

$$\textit{Clean the house} \wedge \neg \textit{Clean the house}.$$

Thus, the task $\alpha \sqcap \beta$ is accomplished if the master requests α and α is accomplished, or the master requests β and β is accomplished. If the master does not make either request, $\alpha \sqcap \beta$ is considered accomplished as there was no concrete task for the slave that it failed to carry out.

In the formalism, making a request is expressed by replacing $\alpha \sqcap \beta$ with α or β. So that in the process of what we can call a *realization* of the task, $\alpha \sqcap \beta$ may evolve to either α, or β, or remain $\alpha \sqcap \beta$. This task thus has at least three possible realizations, represented by the history sequences

$$\langle \alpha \sqcap \beta,\ \alpha,\ \ldots \rangle,$$

$$\langle \alpha \sqcap \beta,\ \beta,\ \ldots \rangle$$

and

$$\langle \alpha \sqcap \beta \rangle.$$

The "..." in the first two sequences indicates that there are passible continuations in case α or β are complex tasks.

The meaning of

$$\bigsqcap x \alpha(x)$$

is the same as the meaning of

$$\alpha(a_0) \sqcap \alpha(a_1) \sqcap \ldots,$$

where $a_0, a_1, \ldots$ are all the objects of the universe of discourse. That is, the task $\bigsqcap x \alpha(x)$ is accomplished if, in the process of realization, the master requests $\alpha(a)$ for one particular object a (replaces $\bigsqcap x \alpha(x)$ with $\alpha(a)$) and $\alpha(a)$ is accomplished, or if the master does not make such a request at all.

Example: Where the universe of discourse is all the targets (geographic locations) on the earth, the task carried out by a transcontinental missile can be expressed by

$$\bigsqcap x Hit(x).$$

The missile (slave) will hit any one particular target the launcher (master) tells it to, but by no means can it hit all the targets, so that it fails to carry out the task

$$\forall x Hit(x).$$

Let us now revisit classical operators for the case when they are applied to tasks containing $\sqcap$ or $\bigsqcap$. For the slave, carrying out $\alpha\wedge\beta$ means carrying out both α and β; for the master, having a slave carrying out $\alpha\wedge\beta$ means the same as having two slaves, one of which carries out α and the other β. So that the master can make any possible requests in α and β in any order, including simultaneous requests. For example, if $\alpha = \alpha_0\sqcap\alpha_1$ and $\beta = \beta_0\sqcap\beta_1$, where $\alpha_0, \alpha_1, \beta_0, \beta_1$ are atomic, the following list shows some of the possible realizations of $\alpha\wedge\beta$:

1. $\langle(\alpha_0\sqcap\alpha_1)\wedge(\beta_0\sqcap\beta_1)\rangle$ (master made no requests);
2. $\langle(\alpha_0\sqcap\alpha_1)\wedge(\beta_0\sqcap\beta_1),\ \alpha_0\wedge(\beta_0\sqcap\beta_1)\rangle$ (master made a request only in α);
3. $\langle(\alpha_0\sqcap\alpha_1)\wedge(\beta_0\sqcap\beta_1),\ (\alpha_0\sqcap\alpha_1)\wedge\beta_1,\ \alpha_0\wedge\beta_1\rangle$ (master first made a request in β and then in α);
4. $\langle(\alpha_0\sqcap\alpha_1)\wedge(\beta_0\sqcap\beta_1),\ \alpha_0\wedge\beta_1\rangle$ (master made a simultaneous request in both α and β).

In case (1), $\alpha\wedge\beta$ is accomplished; in case (2), $\alpha\wedge\beta$ is accomplished iff α_0 is accomplished; and in cases (3) and (4), $\alpha\wedge\beta$ is accomplished iff both α_0 and β_1 are accomplished.

Everything we said about $\alpha\wedge\beta$ also applies to $\alpha\vee\beta$, except that $\alpha\vee\beta$ is considered accomplished if and only if at least one (rather than both) disjunct is eventually accomplished.

The intuitive meaning of the task

$$\alpha\rightarrow\beta$$

is the following: To accomplish $\alpha\rightarrow\beta$ for the slave means to accomplish β as long as α is accomplished under the slave's command. In other words, in the antecedent of the implication, the roles of the master and the slave are interchanged. To the master's requests in β the slave can reply by counterrequests in α. The whole task will be accomplished, if the consequent is eventually accomplished or the antecedent is failed. Notice that the accomplished/failed value of a (sub)task is not relative to whether the (sub)task appears in the antecedent or the consequent of an implication. What is relative is only who commands and in whose "interests" it is to have the (sub)task accomplished or failed.

Example: The agent may not have sufficient resources to unconditionally carry out the task

$$\textit{Clean the house} \sqcap \textit{Clean the lawn},$$

but he may be able to accomplish this task provided that it is given either a mop or a rake, — whichever he requests. Then such an agent can (unconditionally) carry out the task

$$(\textit{Give me a mop} \sqcap \textit{Give me a rake}) \rightarrow (\textit{Clean the house} \sqcap \textit{Clean the lawn}).$$

At any time master can request either cleaning the house or cleaning the lawn. And also at any time the slave can request either giving him a mop or giving him a rake. The task is accomplished if the request of the master (if there was one) is satisfied, or if the request of the slave is not satisfied.

So that one of the possible realizations here is:

1. $(\textit{Give me a mop} \sqcap \textit{Give me a rake}) \rightarrow (\textit{Clean the house} \sqcap \textit{Clean the lawn})$ (the initial state of the task);
2. $(\textit{Give me a mop} \sqcap \textit{Give me a rake}) \rightarrow \textit{Clean the house}$ (master requested to clean the house);
3. $\textit{Give me a mop} \rightarrow \textit{Clean the house}$ (slave requested to give him a mop).

Then the task is accomplished if *Clean the house* has the value *accomplished* (the house has been cleaned), or *Give me a mop* has the value *failed* (a mop has not been given to the slave).

Just as in the case of conjunction, any order of requests is possible in a realization of the above task. The slave could have made a request before the master did (even though such impatience might have been unreasonable of him), or the master and the slave could have made their requests simultaneously.

Generally, a positive occurrence[1] of $\sqcap$ or $\bigsqcap$ signifies master's choice of an action (request), and a negative occurrence signifies slave's choice.

To get a feel of the meaning of tasks with more than one nested $\rightarrow$, let us look at another example:

$$\big((\textit{Open the closet} \sqcap \textit{Open the shed}) \rightarrow (\textit{Give me a mop} \sqcap \textit{Give me a rake})\big)$$
$$\rightarrow (\textit{Clean the house} \sqcap \textit{Clean the lawn}).$$

Here the master may request, say, to clean the house, the slave may reply by the counterrequest to give him a mop, to which the master may reply by the counterrequest to open the closet. The task is then accomplished, if the house has been cleaned, or if a mop has not been given to the slave while the closet has been opened.

The operators $\sqcup$ and $\bigsqcup$ are considered abbreviations defined by

[1] An occurrence is *positive*, if it is in the antecedent of an even number of occurrences of $\rightarrow$, with $\neg\alpha$ understood as $\alpha \rightarrow \bot$. Otherwise the occurrence is *negative*.

- $\alpha \sqcup \beta \;=\; \big((\alpha \to \bot) \sqcap (\beta \to \bot)\big) \to \bot$;
- $\bigsqcup x \alpha(x) \;=\; \big(\bigsqcap x(\alpha(x) \to \bot)\big) \to \bot$.

An agent carrying out $\alpha \sqcup \beta$ should select either α or β and then accomplish the chosen subtask; if no choice is made, then the task is failed. Why this is so can be seen if we disabbreviate $\alpha \sqcup \beta$ and note that $(\alpha \to \bot) \to \bot$ is the same as α, in the sense that an agent can accomplish $(\alpha \to \bot) \to \bot$ if and only if it can accomplish α using the same strategy in the α part of $(\alpha \to \bot) \to \bot$ as it uses in α.

Thus, a positive occurrence of $\sqcup$ signifies slave's choice rather than master's choice as this was the case with $\sqcap$, and a negative occurrence of $\sqcup$, as we may guess, signifies master's choice.

Example: Let P be the task which is accomplished if and only if the woman taking a pregnancy test is pregnant. Then the task carried out by a disposable pregnancy test device can be expressed by $P \sqcup \neg P$. Here not only does the slave "see to" accomplishing one of the tasks P or $\neg P$, but it also effectively tells the master which of them, exactly, is accomplished.

Such a slave is certainly more valuable a resource than a slave accomplishing the task $P \vee \neg P$. The latter is a trivially accomplished task and thus any agent can "accomplish" it. The former task, though, is not that trivial. If it was, then the pregnancy test manufacturers would go bankrupt.

The following two sections contain strict formal definitions of the syntax and the semantics informally introduced in this section. Section 4 introduces an axiomatic system **L** in this formalism, which is shown to be semidecidable in the full language and polynomial space decidable when classical quantifiers are is forbidden in it. This restricted language still contains the quantifier-type operators $\bigsqcap$ and $\bigsqcup$, so its decidability may not be something we would naturally expect from a logic with quantifiers.

6 and 8 contain a proof of the soundness and completeness of **L** in the following intuitive sense: $\mathbf{L} \vdash \alpha$ if and only if α can be carried out by an agent who has nothing but its intelligence for accomplishing tasks. In other words, such an agent has no physical resources or external sources of information to use, and the only way for it to accomplish the task is by means of managing resources implied by the task.

A simple example of a task of this type is $(\alpha \sqcap \beta) \to (\alpha \sqcap \beta)$. If the master requests α (resp. β) in the consequent, the slave requests α (resp. β) in the antecedent. Evidently this strategy guarantees the slave a success.

Not all the formulas that have the form of a classical tautology are valid in this sense though. A counterexample is

$$(\alpha \sqcap \beta) \to \big((\alpha \sqcap \beta) \wedge (\alpha \sqcap \beta)\big).$$

If the master requests α in one of the conjuncts of the consequent and β in the other conjunct, then neither requesting α nor requesting β in the antecedent

by the slave would guarantee that the task is accomplished. Say, if the slave requests β and eventually β is accomplished and α is failed, then the whole task is failed.

This may remind us of linear logic [4] and its variations. They, too, reject the principle $\alpha{\rightarrow}\alpha{\wedge}\alpha$. One can show that every formula that has a form of a theorem of BCK (linear logic + the weakening rule)[2] is also provable in **L**, with the classical operators of the latter understood as multiplicative operators of linear logic, and the operators $\sqcap$ and $\bigsqcap$ understood as the additive conjunction and quantifier, respectively. However, vice versa is not true. For example, every formula of the form

$$\Big(\big(\alpha{\wedge}(\gamma{\sqcap}\delta)\big){\sqcap}\big(\beta{\wedge}(\gamma{\sqcap}\delta)\big){\sqcap}\big((\alpha{\sqcap}\beta){\wedge}\gamma\big){\sqcap}\big((\alpha{\sqcap}\beta){\wedge}\delta\big)\Big){\rightarrow}\big((\alpha{\sqcap}\beta){\wedge}(\gamma{\sqcap}\delta)\big)$$

is provable in **L**, but the above formula is not a theorem of BCK.

Well, so much the worse for linear logic, the main philosophical motivation for introducing which was to have a logic of resources. This claim has never really been supported with a formal semantics, and linear logic owes its name "resource logic" mostly to certain syntactic features (such as the forbidden rule of contraction) rather than a strict and intuitively convincing resource semantics behind it. The author believes that the semantics introduced in this paper has all the chances to be considered an adequate formalization of resource intuition, if resources are understood as agents accomplishing tasks (what is a task for the slave, is a resource for the master). If so, then the fact that linear logic or BCK are weaker than **L** signifies that they are simply incomplete as resource logics.

From the technical point of view, our semantics can be classified as a game semantics, variations of which have been studied by a number of authors over the past 40 years ([1, 2, 6, 8, 12, 13, 3]). With some technical differences in game-playing protocols, common to all those semantics is the idea of considering sentences as games between two players — *proponent*, who is trying to "defend" the sentence, and *opponent*, who is "attacking" the sentence, where proponent makes moves in positive occurrences of disjunction and negative occurrences of conjunction, while opponent makes moves in negative occurrences of disjunction and positive occurrences of conjunction.

Blass [2] was the first to notice the potential of game semantics to justify linear logic, by considering two sorts of conjunction and disjunction in the language, corresponding to two natural sorts of game-playing protocols. This idea was rediscovered by the author of the present paper and further developed in [8], where the decidability of the logic induced by this sort of game semantics was proven (the question left unanswered in [2]), even though, just as in [2], no axiomatization for that logic was found.

Apparently there are two major reasons why the game semantics offered so far have not attracted sufficiently high attention of researchers: either they

[2]BCK is also known under the name Affine Logic. See [2].

seem not very natural and artificially adjusted to some non-semantical considerations, such as justifying certain existing axiomatic systems (intuitionism or linear logic), or totally depart from the original meaning of the language of logic as a formal counterpart of natural language that allows us to talk about the surrounding world), or both.

The author believes that the semantics introduced in this paper does not suffer from the above shortcomings. He has deliberately avoided using game-semantical terms to stress that "this is not just a game".

I also believe that our logic, or certain conservative extensions of it in more expressive languages, have a potential to become an alternative to the existing logics of planning and action used in Artificial Intelligence. The reason for this expectation is that our logic is immune to two major problems most planning logics face: the frame problem and the knowledge preconditions problem. Here is a very brief description of them:

Frame problem: Our actions usually change only a small part of the world and leave unaffected everything else. However, when talking about the effects of an action, we need to explicitly consider not only what this action changes, but also what it does not affect, otherwise we may miss something and get wrong conclusions. This results in dramatic increase in the representational and inferential complexity of planning problems. For more details on the frame problem, see [16].

Knowledge preconditions problem: Most artificial intelligence planners work on the assumption that they have complete knowledge of the world, which, in actual planning situations, is rarely the case. Moreover, certain pieces of knowledge can be acquired from the environment's reactions to certain actions of the planning agent, so that the further action strategy of the agent may have to be contingent on those reactions. The original formalism of the most popular planning logic — situation calculus — fails to capture these nuances. Some researchers ([5, 15]) have approached the problem by extending the language of situation calculus with special means for representing knowledge. This, however, significantly overburdens the formalism, can make things pretty messy, and the corresponding logics typically no longer enjoy the property of semidecidability. McCarthy and Hayes [14] were the first to recognize the knowledge preconditions problem.

When our logic is used for planning, a planning situation is represented as a certain task γ which usually has the form $\alpha{\to}\beta$, where β a description of the goal task for the planning agent (slave) and α is a description of the resources the agent possesses. The knowledge the agent has is nothing but knowledge of the (state of) task γ, so there is no need in having special language or semantical constructs representing knowledge — we don't distinguish and don't really need to distinguish between physical resources and informational resources.

Actions of the planning agent are represented by replacing $\alpha_1{\sqcap}\alpha_2$ (or $\bigsqcap x\alpha(x)$) with α_i (or $\alpha(a)$) in negative parts of γ. These replacements, by the very meaning of this word, only affect the subformulas (resources) in which they are made,

which automatically neutralizes the frame problem.

Reactions of the environment are represented by replacing $\alpha_1 \sqcap \alpha_2$ (or $\bigsqcap x\alpha(x)$) with α_i (or $\alpha(a)$) in positive parts of γ. As the agent has full knowledge of the state of the task it is accomplishing, these reactions are visible to him, so that the knowledge update problem is naturally taken care of.

All this would be better understood on an example. Here is a verbal description of a simple planning problem, followed by a representation of it and a solution in our logic:

There are several sorts of antifreeze coolant available to the agent, and his goal is to

0. Fill the radiator of the car with a safe sort of coolant.

This is what the agent knows:

1. A sort of coolant is safe iff it does not contain acid.

2. A sort of coolant contains acid iff the litmus paper turns red when it is used to test the coolant.

3. At least one of the sorts of coolant: c_1, c_2 is safe.

This is what the agent has or can:

4. A piece of litmus paper which he can use to test any one particular sort of coolant,

and he also can

5. Fill the radiator using any one particular sort of coolant.

Can the agent accomplish the goal described in 0 if all the informational and physical resources he has are the ones described in 1-5, and if yes, how?

Let us try to formalize the problem. First of all, we need to fix a universe of discourse. This would be a certain set of sorts of coolant. c_1 and c_2 are elements of this set.

Next, let us define the following four atomic tasks:

- $A(x)$ that is accomplished iff coolant x contains acid;
- $S(x)$ that is accomplished iff coolant x is safe;
- $R(x)$ that is accomplished iff the litmus paper turns red when used for testing coolant x;
- $F(x)$ that is accomplished iff the radiator has been filled with coolant x.

Then the tasks/resources (0)-(5) can be expressed as follows:

0. $\exists x\big(S(x) \wedge F(x)\big)$
1. $\forall x\big(S(x) \leftrightarrow \neg A(x)\big)$
2. $\forall x\big(A(x) \leftrightarrow R(x)\big)$
3. $S(c_1) \vee S(c_2)$
4. $\bigsqcap x\big(R(x) \sqcup \neg R(x)\big)$
5. $\bigsqcap x F(x)$

To accomplish the goal task (0) having the resources (1)-(5) means nothing but to accomplish the conditional task

$$(1)\wedge(2)\wedge(3)\wedge(4)\wedge(5)\rightarrow(0)$$

unconditionally.

Here is a strategy for carrying out the above task without using any external physical resources or sources of information:

At first, the agent replaces (4) with $R(c_1)\sqcup\neg R(c_1)$. The meaning of this action is using the litmus paper for testing the coolant c_1.

The environment should react by replacing $R(c_1)\sqcup\neg R(c_1)$ with either $R(c_1)$ or $\neg R(c_1)$, for otherwise the task (4) is not accomplished, which would mean that the agent did not really have the resource (accomplishing the task) (4), so that the agent could then "wash his hands". This step corresponds to observing, by the agent, whether the litmus paper turns red or not.

The next action is contingent on what the reaction of the environment to the previous action was.

If the reaction was $R(c_1)$, then the agent replaces (5) with $F(c_2)$. This means having the radiator filled with c_2.

And if the reaction was $\neg R(c_1)$, then the agent replaces (5) with $F(c_1)$, which means having the radiator filled with c_1.

It can be seen that this strategy guarantees success: the radiator will be filled with safe coolant as long as none of the agent's resources (1)-(5) fail to do their job. This strategy will be successful no matter what the meanings of A, S, R, F, c_1, c_2 really are.

Observe how naturally the knowledge preconditions problem was handled without ever mentioning knowledge in the formalism, and how the frame problem was totally out of the scene.

Notice also the convenience of not distinguishing between informational (1-3) and physical (4-5) resources, or between facts (A,S,R) and proper tasks (F). After all, it is not always clear where to draw a line between informational and physical resources, or between facts and proper tasks. E.g., we well might have listed R as a proper task rather than a fact. In any case, as our example demonstrates, there is no need in bothering about drawing those lines.

Thus, the task $(1)\wedge(2)\wedge(3)\wedge(4)\wedge(5)\rightarrow(0)$ can be accomplished by an agent who does not have any resources except those implied by (the antecedent of) the task. By the soundness and completeness theorem for **L**, this is the case if and only if $\mathbf{L}\vdash(1)\wedge(2)\wedge(3)\wedge(4)\wedge(5)\rightarrow(0)$. So that, had our ad hoc methods failed to find an answer, the existence of such an agent (strategy) could have been established by showing that the task we are considering is provable in **L**.

Of course, just knowing that a task-accomplishing strategy exists, would have very little practical value unless we could constructively find such a strategy. No problem: according to Theorem 7.2, such a strategy can be effectively

constructed from an **L**-proof of the task. Moreover, according to Theorem 7.3 which is a stronger version of the soundness theorem for **L**, there is a universal effective strategy that accomplishes every task provable in **L**, i.e. every task that is in principle possible to accomplish.

The main results of the current paper were announced in [11]. A general framework for our approach was first outlined in [9].

Having said all this, it is time to get down to business — formal definitions and proofs.

2 Syntax

We fix a set of expressions that we call *primitive task letters*, with each of which is associated a natural number called its *arity*.

We also fix an infinite set $C = \{c_0, c_1, \ldots\}$ of *constants* and an infinite set $V = \{v_0, v_1, \ldots\}$ of *variables*.

Constants are meant to represent elements of the universe of discourse. For convenience, we are going to simply identify C with the universe of discourse.[3] Requiring that C is infinite does not yield any loss of generality as we shall see later (Fact 7.1).

We refer to elements of $C \cup V$ as *terms*.

For the considerations of compactness of definitions and proofs, we choose a minimal set of basic operators in the language. All the operators not listed in the following definition should be considered standard abbreviations known from classical logic or explained in the Introduction.

Definition 2.1 *Task formulas* are elements of the smallest class of expressions such that:

1. If P is an n-ary primitive task letter and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is a task formula;
2. $\bot$ is a task formula;
3. if α and β are task formulas, then so is $(\alpha) \rightarrow (\beta)$;
4. if α and β are task formulas, then so is $(\alpha) \sqcap (\beta)$;
5. if α is a task formula and x is a variable, then $\forall x(\alpha)$ is a task formula;
6. if α is a task formula and x is a variable, then $\bigsqcap x(\alpha)$ is a task formula.

We will usually use the one-word names "task" or "formula" instead of "task formula". All the three terms should be considered synonyms.

[3] This is the author's favorite approach, first employed in [10].

Throughout the paper lowercase Greek letters (except ϵ) will be used as metavariables for tasks, the letters a and b as metavariables for constants, the letters x and y as metavariables for variables, and the letter t as a metavariable for terms.

Whenever this does not lead to ambiguity, we will omit some parentheses in formulas by standard conventions.

An occurrence of a variable x in a formula is said to be *free*, if it is not in the scope of $\forall x$ or $\bigsqcap x$. Otherwise, the occurrence of x is *bound*. The occurrence of x in $\forall x\alpha$ or $\bigsqcap x\alpha$ is considered bound even if α does not contain x so that x only occurs in the subexpression $\forall x$ or $\bigsqcap x$.

The adjectives "free" and "bound" can be extended to all terms by assuming that an occurrence of a term that happens to be a constant is always free.

A formula α is said to be *safe* if there is no variable that has both a free and a bound occurrence in it. As non-safe formulas can be easily converted into equivalent safe formulas by renaming variables, from now on we restrict our considerations to safe formulas only, and "formula" will always mean "safe formula".

A *closed formula* is a formula that has no free variables.

Where $t_1, \ldots, t_n, t'_1, \ldots, t'_n$ are terms, we will use the expression

$$\alpha(t_1/t'_1, \ldots, t_n/t'_n)$$

to denote the result of replacing in α every free occurrence of t_i $(1 \leq i \leq n)$ by t'_i. When the terms $t_1, \ldots, t_n$ are fixed in the context, we can also simply write

$$\alpha(t'_1, \ldots, t'_n)$$

instead of $\alpha(t_1/t'_1, \ldots, t_n/t'_n)$, as this is an established practice in the literature.

If $x_1, \ldots, x_n$ are all the distinct free variables of α and $a_1, \ldots, a_n$ are constants, then we say that $\alpha(x_1/a_1, \ldots, x_n/a_n)$ is an *instance* of α.

We call the operators $\sqcap$ and $\bigsqcap$ *additive operators*, or *additives*. The other operators from Definition 2.1 will be referred to as *classical operators*.

An *additive (sub)formula* is a (sub)formula of the form $\alpha \sqcap \beta$ or $\bigsqcap x\alpha$. In the former case we say that the additive formula is a $\sqcap$*-formula*, and in the latter case we say that it is a $\bigsqcap$-formula.

The *additive complexity* of a formula is the number of occurrences of additives in that formula.

A *surface occurrence* of a subformula in a formula is an occurrence that is not in the scope of an additive operator.

We need some fixed way to refer to surface occurrences of subformulas. This will be done using strings of the three letters, a (from "antecedent"), c (from "consequent") and q (from "quantifier"). We call this kind of strings *occurrence specifications*. The empty string, denoted by ϵ, is one of the occurrence specifications. Not every occurrence specification is *valid* for a given formula. If Γ is a valid occurrence specification for α, then $\Gamma(\alpha)$ denotes the occurrence in α

specified by Γ, to which we can also refer as "the occurrence Γ in α". Here is a recursive definition of $\Gamma(\alpha)$, where $\vec{r}$ stands for any string over the alphabet $\{a, c, q\}$:

- $a\vec{r}$ is valid for α iff $\alpha = \beta{\rightarrow}\gamma$ and $\vec{r}$ is valid for β, in which case $a\vec{r}(\alpha) = \vec{r}(\beta)$;
- $c\vec{r}$ is valid for α iff $\alpha = \beta{\rightarrow}\gamma$ and $\vec{r}$ is valid for γ, in which case $c\vec{r}(\alpha) = \vec{r}(\gamma)$;
- $q\vec{r}$ is valid for α iff $\alpha = \forall x\beta$ and $\vec{r}$ is valid for β, in which case $q\vec{r}(\alpha) = \vec{r}(\beta)$;
- ϵ is always valid for α and $\epsilon(\alpha) = \alpha$.

Example: Let

$$\alpha = (\bot{\rightarrow}\forall x\sqcap y\gamma) \rightarrow \forall x\big(\sqcap y\gamma{\rightarrow}(\delta{\sqcap}\theta)\big).$$

Then $acq(\alpha)$ is the first occurrence of $\sqcap y\gamma$, $cqa(\alpha)$ is the second occurrence of $\sqcap y\gamma$, $cqc(\alpha)$ is the occurrence of $\delta{\sqcap}\theta$, and aq or $cqcc$ are invalid for α.

An occurrence specification Γ is said to be *positive*, if Γ contains an even (possibly zero) number of a; otherwise it is *negative*. In other words, Γ is positive iff it specifies an occurrence that is in the antecedents of an even number of occurrences of $\rightarrow$.

If Γ is a valid occurrence specification for α and γ is any formula, we call the pair Γ/γ a *replacement* for α.

If $E = \Gamma/\gamma$ is a replacement for α, we will use the expression

$$\alpha(E)$$

or

$$\alpha(\Gamma/\gamma)$$

to denote the result of replacing in α the occurrence Γ by γ.

Definition 2.2

a) An *elementary action* for α is a replacement Γ/γ for α such that Γ is negative and one of the following conditions holds:

- $\Gamma(\alpha) = \beta_0{\sqcap}\beta_1$ and $\gamma = \beta_0$ or $\gamma = \beta_1$; in this case we say that Γ/γ is an *elementary* $\sqcap$*-action* for α.
- $\Gamma(\alpha) = \sqcap x\beta$ and $\gamma = \beta(x/a)$ for some constant a; in this case we say that Γ/γ is an *elementary* $\sqcap$*-action* for α.

b) An *elementary* ($\sqcap$- or $\sqcap$-) *reaction* for α is defined in the same way, only Γ here should be positive rather than negative.

An elementary action means an elementary request made by the slave, and an elementary reaction means an elementary request made by the master.

Definition 2.3 A (simply) *action* for α is a sequence $X = \langle E_1, \ldots, E_n \rangle$ such that E_1 is an elementary action for $\alpha_0 = \alpha$, E_2 is an elementary action for $\alpha_1 = \alpha_0(E_1)$, E_3 is an elementary action for $\alpha_2 = \alpha_1(E_2)$, ...

A (simply) *reaction* for α is defined in the same way, only $E_1, \ldots, E_n$ here should be elementary reactions rather than actions.

When $X = \langle E_1, \ldots, E_n \rangle$ is an action or a reaction for α and $\alpha_0 = \alpha$, $\alpha_1 = \alpha_0(E_1)$, $\alpha_2 = \alpha_1(E_2)$, ..., $\alpha_n = \alpha_{n-1}(E_n)$, we will use the expression

$$\alpha(X)$$

or

$$\alpha(\langle E_1, \ldots, E_n \rangle)$$

to denote α_n, i.e. the result of consecutively making the replacements $E_1, \ldots, E_n$ in α. If here X is the empty action or reaction $\langle\rangle$, then we assume $\alpha(X) = \alpha$.

By abuse of terminology, we will also say that a formula β is an *action* for α, if $\beta = \alpha(X)$ for some action X for α; we say that such an action is *proper*, if $X \neq \langle\rangle$. Similarly, we will say that a formula β is an *elementary* ($\sqcap$- or $\bigsqcap$-) *action* for α, if $\beta = \alpha(X)$ for some elementary ($\sqcap$- or $\bigsqcap$-) action X for α. The same terminological convention applies to reactions and elementary reactions.

The following fact can be easily observed:

Fact 2.4 Assume X is an action and Y is a reaction for α. Then X is also an action for $\alpha(Y)$, and Y is also a reaction for $\alpha(X)$; moreover,

$$\big(\alpha(X)\big)(Y) = \big(\alpha(Y)\big)(X).$$

When X is an action (or a simple action) and Y is a reaction (or a simple reaction) for α, we will usually use the notation

$$\alpha(X, Y)$$

to mean the same as the less symmetric-looking notation $\big(\alpha(X)\big)(Y)$.

If $\beta = \alpha(X, Y)$ for some action X and reaction Y for α, we say that β is a *development* of α; if here $\beta \neq \alpha$, i.e. $X \neq \langle\rangle$ or $Y \neq \langle\rangle$, then β is said to be a *proper development* of α.

Definition 2.5 A *realization* of α is a sequence

$$\langle \alpha_0, \ldots, \alpha_m \rangle$$

such that $\alpha_0 = \alpha$ and for every i with $0 \leq i < m$, α_{i+1} is a proper development of α_i.

When we simply say "a realization", we mean the realization of α for some arbitrary formula α.

Definition 2.6 An *action strategy* is an effective partial function f that assigns to every realization, for which it is defined, an action for the last formula of the realization.

Thus, an action strategy is a procedure that looks at the current state of the task, together with the history of the task, and decides what action to perform (what requests to make) in it.

By abuse of notation, when an action strategy f is undefined for a realization R, we write $f(R) = \langle\rangle$.

Definition 2.7 Let f be an action strategy. A *realization of α with f* is a realization $R = \langle\alpha_0, \ldots, \alpha_m\rangle$ of α such that $f(R) = \langle\rangle$ and for every i with $0 \leq i < m$, if $X = f\langle\alpha_0, \ldots, \alpha_i\rangle$, we have $\alpha_{i+1} = \alpha_i(X, Y)$ for some reaction Y for α_i.

Intuitively, this is how a realization of α_0 with response strategy f is produced: f (the slave) reads the current input $\langle\alpha_0, \ldots, \alpha_i\rangle$ (initially $i = 0$) and starts thinking what action (series of elementary actions) to make for its last formula; while f is thinking, master can make zero, one or more reactions for α, the combination of whose is still called a (one) reaction. Once f has come to a decision, the action it finds is combined with master's reaction(s), applied to the last formula of the input, and the resulting formula α_{i+1} is added to the input as long as it is a proper development of α_i. This way we get a new input (even if it is not different from the previous one), which will again be processed in the same way, and so on.

Thus, our definition allows multiple (non-simple) actions and reactions at every step, as well as simultaneous actions and reactions. The only motivation for this is to get a perfect symmetry between master and slave. An alternative, less symmetric-looking approach, would be to alternate exclusive accesses to the input between master and slave, with or without requiring for requests and responses to be simple. Both variants, as well as any other reasonable variations of the protocol, would produce the same class of valid (accomplishable) tasks. This sort of robustness indicates how natural the semantics is. The situation can be compared with the situation with different sorts of Turing machines and other models of computation that all lead to the same class of computable functions, the phenomenon that serves as a major argument in favor of the Church-Turing thesis.

3 Semantics

A *closed non-logical atom* is $P(a_1, \ldots, a_n)$, where P is an n-ary primitive task letter and each a_i is a constant.

Definition 3.1 An *eventual situation* is a total function s that assigns to every closed non-logical atom one of the values {1,0}. 1 corresponds to the intuitive value *accomplished*, and 0 corresponds to the value *failed*.

This function is extended to all closed formulas as follows:

1. $s(\bot) = 0$.
2. $s(\alpha{\rightarrow}\beta) = \begin{cases} 0 & \text{if } s(\alpha) = 1 \text{ and } s(\beta) = 0; \\ 1 & \text{otherwise.} \end{cases}$
3. $s(\forall x\alpha) = \begin{cases} 1 & \text{if } s\big(\alpha(x/a)\big) = 1 \text{ for every constant } a; \\ 0 & \text{otherwise.} \end{cases}$
4. $s(\alpha{\sqcap}\beta) = 1$.
5. $s(\bigsqcap x\alpha) = 1$.

A task is said to be *primitive*, if its additive complexity is 0, i.e. it does not contain any additive operators.

The *primitivization* of α, denoted by $\overline{\alpha}$, is the result of replacing in α all (surface) occurrences of all additive subformulas by $\top$, which abbreviates $\bot{\rightarrow}\bot$.

The following fact immediately follows from Definition 3.1:

Fact 3.2 Assume α is a closed formula. Then, for every eventual situation s, we have $s(\alpha) = s(\overline{\alpha})$.

An eventual situation s can be thought of as a classical model, where the universe of discourse is C (the set of all constants), and where an n-ary primitive task letter P is interpreted as the n-ary predicate that is true for the n-tuple $a_1, \ldots, a_n$ of constants if and only if $s\big(P(a_1, \ldots, a_n)\big) = 1$.

When restricted to primitive formulas, Definition 3.1 is virtually the same as the classical definition of truth in model s. So that, for every primitive formula α, we have $s(\alpha) = 1$ iff α is true in s in the classical sense. In view of the soundness and completeness of classical first order predicate calculus, we then have:

Fact 3.3 A primitive formula α is provable in classical first order logic iff for every eventual situation s, we have $s(\alpha) = 1$.

Observe that development preserves the ($\rightarrow$,$\forall$)-structure of the formula. I.e., a development of $\alpha{\rightarrow}\beta$ always has the form $\alpha'{\rightarrow}\beta'$, where α' and β' are developments of α and β, respectively. Similarly, a development of $\forall x\alpha$ always has the form $\forall x\alpha'$, where α' is a development of α.

It follows that if an occurrence specification is valid for a formula, it is also valid for any development of that formula.

Assume $R = \langle\alpha_0, \ldots, \alpha_n\rangle$ is a realization of α_0 and Γ is a valid occurrence specification for α_0. Then the *projection of R on Γ*, denoted by $\Gamma(R)$, is the

result of deleting in $\langle \Gamma(\alpha_0), \ldots, \Gamma(\alpha_n) \rangle$ every formula that is a duplicate of its predecessor in the sequence. It is obvious that $\Gamma(R)$ is a realization of $\Gamma(\alpha_0)$.

We also extend the notation $\alpha(t/t')$ to realizations: If R is a realization and t, t' are terms, $R(t/t')$ denotes the result of replacing all free occurrences of t by t' in every formula of R.

Definition 3.4 We say that a realization $R = \langle \alpha_0, \ldots, \alpha_m \rangle$ of a closed formula α_0 is *successful* with respect to an eventual situation s, if one of the following conditions holds:

- α_0 is atomic (which implies $m = 0$) and $s(\alpha_0) = 1$;
- $\alpha_0 = \beta \rightarrow \gamma$ and $c(R)$ is successful with respect to s whenever $a(R)$ is so;
- $\alpha_0 = \forall x \beta$ and, for every constant a, $\big(q(R)\big)(x/a)$ is successful with respect to s;
- α_0 is an additive formula and either $m = 0$ or $m \geq 1$ and $\langle \alpha_1, \ldots, \alpha_m \rangle$ is successful with respect to s.

This definition formalizes the intuition on accomplishing a task that was described in the Introduction. In the context of an actual realization $\langle \alpha_0, \ldots, \alpha_m \rangle$, the task α_0 should be considered accomplished if the realization is successful with respect to the (actual) eventual situation. In particular, an atomic task is accomplished iff it has the value 1 in the eventual situation. The task $\beta \rightarrow \gamma$ is accomplished iff γ (the projection of the realization on the consequent) is accomplished as long as β (the projection on the antecedent) is accomplished. The task α_0 of the form $\beta \sqcap \gamma$ or $\bigsqcap x \beta$ is accomplished iff either there was no request specifying which particular subtask implied by α_0 should be accomplished ($m = 0$), or there was such a request ($m \geq 1$) and the requested subtask was accomplished. In particular, this subtask is α_1, which would be either β or γ if $\alpha_0 = \beta \sqcap \gamma$, and would be $\beta(a)$ if $\alpha_0 = \bigsqcap x \beta(x)$.

As for $\forall x \beta(x)$, the Introduction did not contain very clear explanation of what it means to accomplish this task. It was only mentioned that accomplishing $\forall x \beta(x)$ means accomplishing $\beta(x)$ "for all particular values of x". A possible interpretation of this statement could be that the meaning of $\forall x \beta(x)$ is the same as the meaning of $\beta(c_0) \wedge \beta(c_1) \wedge \ldots$ (all constants c_i). I have reserved a different symbol, $\bigwedge$, for this version of interpretation of "for all". The semantics of $\bigwedge x \beta(x)$ will be discussed in more detail in Section 9. As to $\forall x \beta(x)$, Definition 3.4 implies a different intuition for it: Accomplishing $\forall x \beta(x)$ means having $\beta(x)$ accomplished for any possible particular value of x, without knowing this particular value. The slave should act in a way that guarantees that $\beta(x)$ is accomplished no matter what the value of x is. Thus, unlike $\bigwedge x \beta(x)$ which produces infinitely many subtasks $\beta(c_0), \beta(c_1), \ldots$ for the slave to accomplish simultaneously, $\forall x \beta(x)$ only produces one task. In this respect, $\forall x \beta(x)$ is similar to $\bigsqcap x \beta(x)$ which, too, signifies only one task for the slave to accomplish.

The difference between $\sqcap x\beta(x)$ and $\forall x\beta(x)$ is that in the former the slave is explicitly told for which particular value of x it has to accomplish $\beta(x)$, while in the latter this value remains unspecified.

This operator is useful in modelling "black box" type of problems: there is an object x in the black box and the agent needs to accomplish a certain task involving this object without knowing what particular object it is.

Example: The telephone memory (black box) stores the number (object in the box) last dialed from this phone; the agent, who does not happen to know the value of that number, still can accomplish the task of dialing this number by hitting the "redial" button.

Lemma 3.5 A realization of a closed task is successful with respect to an eventual situation s if and only if s assigns the value 1 to the last formula of the realization.

PROOF Assume R is a realization of a closed task α and s is an eventual situation. Below "successful" means "successful in s".

We use induction on the complexity of α.

If α is atomic, then, by Definition 3.4, R is successful with respect to s iff $s(\alpha) = 1$. It remains to notice that $R = \langle\alpha\rangle$, so that α is (the first and) the last formula of R.

Assume $\alpha = \beta_0{\rightarrow}\gamma_0$. Then R has the form $\langle\beta_0{\rightarrow}\gamma_0, \ldots, \beta_m{\rightarrow}\gamma_m\rangle$. Therefore β_m is the last formula of $a(R)$ as the latter is nothing but $\langle\beta_0 \ldots \beta_m\rangle$ with duplicate formulas removed. Similarly, γ_m is the last formula of $c(R)$. $a(R)$ and $c(R)$ are realizations of β_0 and γ_0, respectively. The complexities of β_0 and γ_0 are lower than that of α. Hence, by the induction hypothesis, $a(R)$ is successful iff $s(\beta_m) = 1$, and $c(R)$ is successful iff $s(\gamma_m) = 1$. Therefore, in view of the clause 2 of Definition 3.1, $s(\beta_m{\rightarrow}\gamma_m) = 1$ iff $a(R)$ is not successful or $c(R)$ is successful. But, by Definition 3.4, this is the case if and only if R is successful. Thus, R is successful iff s assigns 1 to its last formula $\beta_m{\rightarrow}\gamma_m$.

Assume now $\alpha = \forall x\beta_0$. Then R has the form $\langle\forall x\beta_0, \ldots, \forall x\beta_m\rangle$. Therefore, for a constant a, $\beta_m(x/a)$ is the last formula of $\big(q(R)\big)(x/a)$. The latter is a realization of $\beta_0(x/a)$, whose complexity is lower than that of α. Hence, by the induction hypothesis, $\big(q(R)\big)(x/a)$ is successful iff $s\big(\beta_m(x/a)\big) = 1$. Therefore, in view of the clause 3 of Definition 3.1, $s(\forall x\beta_m) = 1$ iff $\big(q(R)\big)(x/a)$ is successful for every constant a. But, by Definition 3.4, this is the case if and only if R is successful. Thus, again, R is successful iff s assigns 1 to its last formula $\forall x\beta_m$.

Finally, assume α is $\beta{\sqcap}\gamma$ or $\sqcap x\beta$. If $R = \langle\alpha\rangle$, then, by Definition 3.4, R is successful and, by Definition 3.1, $s(\alpha) = 1$. If $R = \langle\alpha, \alpha_1, \ldots, \alpha_m\rangle$ $(m \geq 1)$, then α_1 must be a proper development of α and hence have a lower complexity than α does. Therefore, by the induction hypothesis, the realization $\langle\alpha_1, \ldots, \alpha_m\rangle$ of α_1 is successful iff $s(\alpha_m) = 1$. By Definition 3.4, R is successful iff $\langle\alpha_1, \ldots, \alpha_m\rangle$ is so. Thus, R is successful iff $s(\alpha_m) = 1$. $\Box$

The reader may ask why I have not chosen Lemma 3.5 as a definition of "successful" instead of Definition 3.4. There are two reasons: First of all, it is Definition 3.4 rather than Lemma 3.5 that captures the task intuition described in the Introduction. Secondly, the portability of the definition would be lost. The point is that Definition 3.4 has natural generalizations for certain more expressive languages, — languages where realizations may no longer be finite which would make Lemma 3.5 inapplicable. Some languages of this sort are described in Section 9.

Definition 3.6 We say that an action strategy f *accomplishes* α if, for every instance α^* of α and every eventual situation s, every realization of α^* with f is successful with respect to s.

If such an action strategy f exists, we say that α is *accomplishable*.

One can verify that accomplishability of a task α containing free variables is equivalent to accomplishability of the $\sqcap$-closure of α rather than its $\forall$-closure. That is, if $x_1, \ldots, x_n$ are all the free variables of α, then α is accomplishable iff $\sqcap x_1 \ldots \sqcap x_n \alpha$ is accomplishable.

Definition 3.6 is a formalization of the intuitive meaning of accomplishability as existence of an agent who has an effective action strategy that guarantees that the task will be accomplished no matter what the environment's reactions are and what the eventual situation is. This agent has no physical or informational resources beyond those implied by the task.

There is one important point worth noticing. Technically, the definition of accomplishability is similar to the classical definition of validity (tautology): they both involve universal quantification over situations/models. In classical logic, however, the point of departure is *truth* rather than validity, and validity is interesting only as long as it means truth in every particular model. This is so because every application of classical logic deals with one fixed model, and what has practical significance is whether a given fact is true in this particular model rather than in every possible model. As for the logic of tasks, the situation with it is different. Here the central semantical concept in its own rights is accomplishability rather than (a counterpart of) truth. Speaking about one particular actual eventual situation does not make practical sense — this situation is not initially determined, and which one of the possible situations becomes "eventual" may depend on the actions of the agent. The job of the agent is to influence the course of events in a way that narrows down the set of all potential eventual situations to favorable ones.

And even if some of the primitive task letters of the language stand for facts (tasks with predetermined semantical values, — see the Introduction), so that the eventual situation is partially predetermined, it is still accomplishability rather than truth that is of primary significance. Because what matters for successful planning (the main practical application of the logic of tasks) is not whether the semantical values of primitive tasks are predetermined or not,

but whether and how much the agent knows about those values. The coolant problem discussed in the Introduction is a good demonstration of this point.

4 Logic L

The logic **L** that we are going to define in this section, is intended to axiomatize the set of accomplishable formulas. Its soundness and completeness with respect to accomplishability will be proven in later sections.

We will say that the variable v_i is *smaller* than the variable v_j, if $i < j$.

Throughout the paper we will be using the expression

$$x_\alpha$$

to denote the smallest variable that has no (free or bound) occurrence in α.

We will also be using the expression

$$\text{Freeterms}(\alpha)$$

for the set of all terms (variables or constants) that have a free occurrence in α.

Definition 4.1

a) An *elementary quasiaction* for α is a replacement Γ/γ for α such that Γ is negative and one of the following conditions holds:

1. $\Gamma(\alpha) = \beta_0 \sqcap \beta_1$ and $\gamma = \beta_0$ or $\gamma = \beta_1$. In this case we say that Γ/γ is an *elementary* $\sqcap$*-quasiaction* for α.

2. $\Gamma(\alpha) = \bigsqcap x\beta$ and $\gamma = \beta(x/t)$, where $t \in \text{Freeterms}(\alpha)$ or $t = x_\alpha$. In this case we say that Γ/γ is an *elementary* $\bigsqcap$*-quasiaction* for α.

b) An *elementary quasireaction* for α is a replacement Γ/γ for α such that Γ is positive and one of the following conditions holds:

1. $\Gamma(\alpha) = \beta_0 \sqcap \beta_1$ and $\gamma = \beta_0$ or $\gamma = \beta_1$. In this case we say that Γ/γ is an *elementary* $\sqcap$*-quasireaction* for α.

2. $\Gamma(\alpha) = \bigsqcap x\beta$ and $\gamma = \beta(x/x_\alpha)$. In this case we say that Γ/γ is an *elementary* $\bigsqcap$*-quasireaction* for α.

Note the similarity and the difference between elementary action (reaction) and elementary quasiaction (quasireaction). At the propositional level there is no difference. The difference shows up when we deal with formulas containing $\bigsqcap$. Every formula only has a finite number of elementary quasiactions or quasireactions, while the number of elementary actions or reactions for a formula can be infinite. Notice also that Definition 4.1 no longer enjoys the perfect symmetry of Definition 2.2 between elementary actions and elementary reactions.

Now we are ready to define logic **L**.

Definition 4.2 of LOGIC L

- The **axioms** are all the primitive formulas provable in classical first order logic.
- The **rules of inference** are:

  Rule A: $\dfrac{\pi}{\alpha}$,

  where π is an elementary quasiaction for α;

  Rule R: $\dfrac{\overline{\alpha}, \pi_1, \ldots, \pi_e}{\alpha}$,

  where $e \geq 1$ and $\pi_1, \ldots, \pi_e$ are all the elementary quasireactions for α.

We adopt the linear rather than tree-like version of proofs: $\mathbf{L} \vdash \alpha$ ("α is *provable* in $\mathbf{L}$") means that there is a finite sequence of formulas ending with α, where each formula is either an axiom, or follows by one of the rules of inference from some earlier formulas in the sequence. The number of formulas in such a sequence is called the *length* of the proof.

We use the expression

$$\mathbf{L} \vdash_l \alpha$$

to say that α has a proof in $\mathbf{L}$ whose length is at most l.

5 The decidability of the $\forall$-free fragment of L

Let $\mathbf{L}^-$ be $\mathbf{L}$ restricted to formulas not containing $\forall$. $\mathbf{L}^-$ is more than the propositional fragment of $\mathbf{L}$, because its language still has $\bigsqcap$.

Theorem 5.1

1. $\mathbf{L}$ is semidecidable.
2. $\mathbf{L}^-$ is decidable. In particular, it is decidable in polynomial space.

PROOF The clause 1 is obvious in view of the semidecidability of classical first order logic and the way $\mathbf{L}$ is defined.

As for the clause 2, here is an informal description of a decision procedure for $\mathbf{L}^- \vdash \alpha$, together with a proof, by induction on the additive complexity of α, that the procedure takes a finite time.

Given a formula α,

a) If α is primitive, then the rules A and R are not applicable to α, and the only way α can be derived in $\mathbf{L}^-$ is if it is an axiom of $\mathbf{L}^-$, i.e. if it is provable in classical logic. Since α does not contain quantifiers, it is provable in classical logic if and only if it is a classical propositional tautology. So, check whether α is a tautology, and if it is, output "yes", otherwise output "no". In view of

the decidability of classical propositional logic, this step can be completed in a finite time.

b) If α is not primitive, then the only way it can be proved in $\mathbf{L}^-$ is if either one of the elementary quasiactions for it is provable, or all of the elementary quasireactions for it, together with its primitivization, are provable in $\mathbf{L}^-$. The primitivization is provable iff it is a tautology, so its provability can be checked in a finite time. Also, as we noted, the number all the elementary quasiactions and quasireactions for α is finite. So, check each of them for provability in $\mathbf{L}^-$. If it turns out that either one of the elementary quasiactions, or all of the elementary quasireactions together with the primitivization of α are provable in $\mathbf{L}^-$, then output "yes", otherwise output "no". The additive complexities of those elementary quasiactions and quasireactions are lower than the additive complexity of α and, by the induction hypothesis, their provability in $\mathbf{L}$ can be checked in a finite time. So that this step, too, can be completed in a finite time.

A more careful analysis of the above procedure could convince us that the procedure takes (at most) polynomial space. $\Box$

6 The soundness of L

Lemma 6.1 For any terms $t_1, \ldots, t_n, t'_1, \ldots, t'_n$, where $t'_1, \ldots, t'_n$ have no bound occurrences in α,

$$\text{if } \mathbf{L} \vdash_l \alpha, \text{ then } \mathbf{L} \vdash_l \alpha(t_1/t'_1, \ldots, t_n/t'_n).$$

PROOF

Assume $\mathbf{L} \vdash_l \alpha$. Let

$$\alpha' = \alpha(t_1/t'_1, \ldots, t_n/t'_n).$$

We use induction on l to show that $\mathbf{L} \vdash_l \alpha'$.

If l=1, i.e. α is an axiom, then α is provable in classical logic. The latter is known to be closed under free substitution for variables and constants, so classical logic proves $\alpha(t_1/t'_1, \ldots, t_n/t'_n)$. Consequently, $\mathbf{L} \vdash_1 \alpha(t_1/t'_1, \ldots, t_n/t'_n)$.

Suppose α is obtained from π by the A-rule. Then $\mathbf{L} \vdash_{l-1} \pi$, whence, by the induction hypothesis, $\mathbf{L} \vdash_{l-1} \pi'$, where

$$\pi' = \pi(x_\alpha/x_{\alpha'}, t_1/t'_1, \ldots, t_n/t'_n).$$

It is easy to verify that since π is an elementary quasiaction for α, π' is an elementary quasiaction for α'. Therefore for α', which follows from π' by the A-rule, we have $\mathbf{L} \vdash_l \alpha'$.

Finally, suppose α is obtained from $\overline{\alpha}, \pi_1, \ldots, \pi_e$ by the R-rule.

By the induction hypothesis, for each i with $1 \leq i \leq e$, the formula

$$\pi_i' = \pi_i(x_\alpha / x_{\alpha'}, t_1/t_1', \ldots, t_n/t_n')$$

has a proof of the same length as π_i does. One can verify that

$$\pi_1',\ \ldots,\ \pi_e'$$

are all the quasireactions for α', and $\overline{\alpha}(t_1/t_1', \ldots, t_n/t_n')$ is the primitivization of α'. So that α' follows from

$$\overline{\alpha}(t_1/t_1', \ldots, t_n/t_n'),\ \pi_1',\ \ldots,\ \pi_e'$$

by the R-rule. Applying the induction hypothesis to the premises of this rule, we conclude that α' has a proof of the same length as α does. $\Box$

Lemma 6.2 If $\mathbf{L} \vdash_l \alpha$ and ρ is an elementary quasireaction for α, then $\mathbf{L} \vdash_{l-1} \rho$.

PROOF We use induction on l.

If $l = 1$, i.e. α is an axiom, then α has no elementary quasireactions and we are done.

Assume now $l > 1$ and

$$\rho = \alpha(\Delta/\delta)$$

is an elementary quasireaction for α.

If α is obtained by the R-rule from $\overline{\alpha}, \pi_1, \ldots, \pi_e$, then $\rho = \pi_i$ for one of the i with $1 \leq i \leq e$. Each of these π_i has a proof of length $\leq (l-1)$, and so does ρ.

Suppose now α is obtained by the A-rule from an elementary quasiaction

$$\pi = \alpha(\Gamma/\gamma)$$

for α.

To show that ρ has a proof of length $\leq (l-1)$, we need to consider four cases, depending on whether Γ/γ and Δ/δ are elementary $\sqcap$- or $\bigsqcap$-quasiaction/quasireaction.

Below we will be using the following readability-improving notational convention: for any formulas ϕ and ψ,

$$\alpha(\Gamma/\phi,\ \Delta/\psi)$$

will be abbreviated as

$$\alpha[\phi, \psi].$$

Case 1: $\Gamma(\alpha) = \xi_0 \sqcap \xi_1$ and $\Delta(\alpha) = \eta_0 \sqcap \eta_1$, with $\gamma = \xi_i$ and $\delta = \eta_j$ $(i, j \in \{0, 1\})$. Thus we have:

$$\alpha = \alpha[\xi_0 \sqcap \xi_1,\ \eta_0 \sqcap \eta_1],$$

$$\pi = \alpha[\xi_i,\ \eta_0 \sqcap \eta_1]$$

and

$$\rho = \alpha[\xi_0 \sqcap \xi_1,\ \eta_j].$$

Let

$$\theta = \alpha[\xi_i,\ \eta_j].$$

Obviously θ is an elementary quasireaction for π and an elementary quasiaction for ρ. That θ is an elementary quasireaction for π implies, by the induction hypothesis, that θ has a shorter proof than π does, so that, since $\mathbf{L} \vdash_{l-1} \pi$, we have $\mathbf{L} \vdash_{l-2} \theta$. Consequently, since ρ can be derived from θ by the A-rule, we have $\mathbf{L} \vdash_{l-1} \rho$.

Case 2: $\Gamma(\alpha) = \xi_0 \sqcap \xi_1$ with $\gamma = \xi_i$ $(i \in \{0, 1\})$, and $\Delta(\alpha) = \bigsqcap y\eta$ with $\delta = \eta(y/x_\alpha)$. So that we have:

$$\alpha = \alpha[\xi_0 \sqcap \xi_1,\ \bigsqcap y\eta(y)],$$

$$\pi = \alpha[\xi_i,\ \bigsqcap y\eta(y)]$$

and

$$\rho = \alpha[\xi_0 \sqcap \xi_1,\ \eta(x_\alpha)].$$

Let

$$\theta = \alpha[\xi_i,\ \eta(x_\alpha)]$$

and

$$\theta' = \alpha[\xi_i,\ \eta(x_\pi)].$$

Obviously θ' is an elementary quasireaction for π and hence, by the induction hypothesis (as $\mathbf{L} \vdash_{l-1} \pi$), we have $\mathbf{L} \vdash_{l-2} \theta'$. Consequently, by Lemma 6.1,

$$\mathbf{L} \vdash_{l-2} \theta'(x_\pi/x_\alpha).$$

But observe that $\theta'(x_\pi/x_\alpha) = \theta$. Thus, $\mathbf{L} \vdash_{l-2} \theta$. It remains to notice that θ is an elementary quasiaction for ρ; applying the A-rule to θ, we then get $\mathbf{L} \vdash_{l-1} \rho$.

Case 3: $\Gamma(\alpha) = \bigsqcap x\xi$ with $\gamma = \xi(x/t)$, where $t = x_\alpha$ or $t \in \text{Freeterms}(\alpha)$, and $\Delta(\alpha) = \eta_0 \sqcap \eta_1$ with $\delta = \eta_j$ $(j \in \{0, 1\})$, so that we have:

$$\alpha = \alpha[\bigsqcap x\xi(x),\ \eta_0 \sqcap \eta_1],$$

$$\pi = \alpha[\xi(t),\ \eta_0 \sqcap \eta_1]$$

and

$$\rho = \alpha[\bigsqcap x\xi(x),\ \eta_j].$$

Let

$$\theta = \alpha[\xi(t),\ \eta_j)].$$

Obviously θ is an elementary quasireaction for π, so that, by the induction hypothesis (as $\mathbf{L} \vdash_{l-1} \pi$), $\mathbf{L} \vdash_{l-2} \theta$. If $t \in \text{Freeterms}(\rho)$, then clearly θ is also an elementary quasiaction for ρ, whence, by the A-rule, we get $\mathbf{L} \vdash_{l-1} \rho$.

Suppose now $t \notin \text{Freeterms}(\rho)$. Let then

$$\theta' = \alpha[\xi(x_\rho),\ \eta_j].$$

Observe that t has no free or bound occurrences in θ' (otherwise t would be either in Freeterms(ρ), or among the bound variables of α). Therefore it is easy to see that $\theta' = \theta(t/x_\rho)$. Since we have $\mathbf{L} \vdash_{l-2} \theta$, Lemma 6.1 implies that $\mathbf{L} \vdash_{l-2} \theta'$. But θ' is an elementary quasireaction for ρ so that the latter follows from the former by the A-rule. Consequently, $\mathbf{L} \vdash_{l-1} \rho$.

Case 4: $\Gamma(\alpha) = \bigsqcap x\xi$ with $\gamma = \xi(x/t)$, where $t = x_\alpha$ or $t \in \text{Freeterms}(\alpha)$, and $\Delta(\alpha) = \bigsqcap y\eta$, with $\delta = \eta(y/x_\alpha)$, so that we have:

$$\alpha = \alpha[\bigsqcap x\xi(x),\ \bigsqcap y\eta(y)],$$

$$\pi = \alpha[\xi(t),\ \bigsqcap y\eta(y)]$$

and

$$\rho = \alpha[\bigsqcap x\xi(x),\ \eta(x_\alpha)].$$

Let

$$\theta = \alpha[\xi(t),\ \eta(x_\pi)].$$

θ is an elementary quasireaction for π and, as $\mathbf{L} \vdash_{l-1} \pi$, by the induction hypothesis, we have

$$\mathbf{L} \vdash_{l-2} \theta. \tag{1}$$

We need to consider two subcases:

Subcase 4.1: $t \in \text{Freeterms}(\alpha)$.

Then clearly we also have $t \in \text{Freeterms}(\rho)$ and therefore the formula θ' defined by

$$\theta' = \alpha[\xi(t),\ \eta(x_\alpha)]$$

is an elementary quasiaction for ρ. One can verify that $\theta' = \theta(x_\pi/x_\alpha)$ which, by Lemma 6.1 together with (1), implies $\mathbf{L} \vdash_{l-2} \theta'$. Hence, as ρ can be obtained from θ' by the A-rule, we have $\mathbf{L} \vdash_{l-1} \rho$.

Subcase 4.2: $t = x_\alpha$.

Let then

$$\theta' = \alpha[\xi(x_\rho),\ \eta(x_\alpha)].$$

One can verify that $\theta' = \theta(t/x_\rho, x_\pi/x_\alpha)$. This, by Lemma 6.1 together with (1), implies that $\mathbf{L} \vdash_{l-2} \theta'$. But θ' is an elementary quasiaction for ρ. Remembering about the A-rule, we can conclude that $\mathbf{L} \vdash_{l-1} \rho$. $\square$

We say that β is a (simply) *quasiaction* for α if $\beta = \alpha$, or β is an elementary quasiaction for α, or β is an elementary quasiaction for an elementary quasiaction for α, or ... β is said to be a *proper* quasiaction for α, if β is a quasiaction for α and $\beta \neq \alpha$.

Similarly, *quasireaction* is the transitive and reflexive closure of the relation "elementary quasireaction", and *proper quasireaction* is the transitive closure of that relation.

It is easy to see that just as this is the case with elementary quasiactions and elementary quasireactions, the number of all quasiactions and quasireactions for any given formula is finite.

Lemma 6.3 If β^* is a reaction (resp. proper reaction) for an instance of α, then β^* is an instance of a quasireaction (resp. proper quasireaction) for α.

PROOF Let α^* be an instance of α and $\beta^* = \alpha^*(Y)$ be a reaction for α^*.

We prove the lemma by induction on the length of Y.

Basis: If $Y = \langle\rangle$, i.e. the reaction Y is not proper, then $\beta^* = \alpha^*$. As α is a (non-proper) quasireaction for itself and α^* is an instance of it, we are done.

Inductive step: Assume $Y = \langle E_1, \ldots, E_k, \Gamma/\gamma^* \rangle$ $(k \geq 0)$. Then Y is a proper reaction and $\alpha^*(Y) = \theta^*(\Gamma/\gamma^*)$, where $\theta^* = \alpha^*(\langle E_1, \ldots, E_k \rangle)$. By the induction hypothesis, θ^* is an instance of a quasireaction θ for α. So, it suffices to show that $\theta^*(\Gamma/\gamma^*)$ is an instance of an elementary quasireaction for θ and hence of a proper quasireaction for α.

There are two cases to consider:

Case 1: Γ/γ^* is an elementary $\sqcap$-reaction for α^*, i.e. $\Gamma(\alpha^*) = \beta_0^* \sqcap \beta_1^*$ and γ^* is one of the β_i^*, $i \in \{0, 1\}$. Fix this i.

Since $\Gamma(\theta^*) = \beta_0^* \sqcap \beta_1^*$ and θ^* is an instance of θ, we must have $\Gamma(\theta) = \beta_0 \sqcap \beta_1$, where $\beta_0^* \sqcap \beta_1^*$ is an instance of $\beta_0 \sqcap \beta_1$. Then Γ/β_i is an elementary quasireaction for θ, and obviously $\theta^*(\Gamma/\beta_i^*)$ is an instance of $\theta(\Gamma/\beta_i)$.

Case 2: Γ/γ^* is an elementary $\bigsqcap$-reaction for α^*, i.e. $\Gamma(\alpha^*) = \bigsqcap y \beta^*$ and γ^* is $\beta^*(y/b)$ for some constant b. Fix this constant.

Since $\Gamma(\theta^*) = \bigsqcap y \beta^*$ and θ^* is an instance of θ, we must have $\Gamma(\theta) = \bigsqcap y \beta$, where β^* is an instance of β. Then $\Gamma/\beta(y/x_\theta)$ is an elementary quasireaction for θ, and obviously $\theta^*\big(\Gamma/\beta^*(x/b)\big)$ is an instance of $\theta\big(\Gamma/\beta(x/x_\theta)\big)$. $\Box$

Theorem 6.4 (Soundness of L) If $\mathbf{L} \vdash \alpha$, then α is accomplishable.

PROOF Assume $\mathbf{L} \vdash_l \alpha$. Let

$$x_1, \ldots, x_n$$

be all the free variables of α.

Below we define an action strategy f and show, by induction on l, that it accomplishes (any instance of) α.

Case 1: Assume α is an axiom, i.e. $l = 1$. This implies that α is a primitive formula provable in classical logic, and so are all instances of α.

Let in this case f be the action strategy that assigns $\langle\rangle$ to every realization. f can be called the "idle strategy". Consider an arbitrary instance $\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$ of α. As α^* is primitive, there are no nonempty reactions for it, and $\langle\alpha^*\rangle$ is the only possible realization of α^* with f. Therefore, in view of Lemma 3.5, $\langle\alpha^*\rangle$ is successful with respect to an eventual situation s iff $s(\alpha^*) = 1$. But since α^* is provable in classical logic, by Fact 3.3, for every eventual situation s, we have $s(\alpha^*) = 1$. Thus, f accomplishes every instance α^* of α, i.e. f accomplishes α. Notice that in fact any action strategy accomplishes α.

Case 2: Assume α follows from π by the A-rule. The proof of π is shorter than l and, by Lemma 6.2, every quasireaction for π also has a proof shorter than l. Let $\beta_1, \ldots, \beta_k$ be all the quasireactions for π. By the induction hypothesis, there are action strategies

$$f_1, \ldots, f_k$$

that accomplish $\beta_1, \ldots, \beta_k$, respectively.

Since α follows from π by the A-rule, we must have $\pi = \alpha(\Gamma/\gamma)$ for some elementary quasiaction Γ/γ for α. There are two subcases to consider:

Subcase 2.1: Assume $\Gamma(\alpha) = \xi_0 \sqcap \xi_1$ and $\gamma = \xi_i$ for one of the $i \in \{0, 1\}$. Fix this i. Let then f be the function that, for every instnce

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

of α, acts as follows:

- $f\langle\alpha^*\rangle = X$, where $X = \langle\Gamma \ / \ \xi_i(x_1/a_1/ \ldots, x_n/a_n)\rangle$;
- For any realization $\langle\alpha^*, \alpha_1, \ldots, \alpha_m\rangle$ with $m \geq 1$ where α_1 is an instance of one of the β_j $(1 \leq j \leq k)$, f returns the same value as f_j does for the realization $\langle\alpha_1, \ldots, \alpha_m\rangle$. If here there are more than one such β_j, the one with the smallest j is selected.

Observe that X is indeed an action for α^* and $\alpha^*(X)$ is an instance of π. In particular, $\alpha^*(X) = \pi(x_1/a_1, \ldots, x_n/a_n)$.

We claim that f accomplishes α. To see this, consider an arbitrary instance $\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$ of α. Since X is nonempty, every realization of α^* with f has the form $\langle\alpha^*, \alpha^*(X, Y), \theta_1, \ldots, \theta_m\rangle$ $(m \geq 0)$, where Y is a reaction for α^* and hence for $\alpha^*(X)$. As we noted, the latter is an instance of π. Therefore, by Lemma 6.3, $\alpha^*(X, Y)$ is an instance of a quasireaction for π. Remember that $\beta_1, \ldots, \beta_k$ are all the quasireactions for π, so $\alpha^*(X, Y)$ must be an instance of one of them. Let j be the smallest number such that $\alpha^*(X, Y)$ is an instance of β_j. Thus, the realization of α^* has the form $\langle\alpha^*, \beta_j^*, \theta_1, \ldots, \theta_m\rangle$, where β_j^* is an instance of β_j. From the way f is defined, it is easy to see that then

$\langle \beta_j^*, \theta_1, \ldots, \theta_m \rangle$ is a realization of β_j^* with the action strategy f_j. It was our assumption that the latter accomplishes β_j and hence β_j^*. Therefore, by Lemma 3.5, the last formula of $\langle \beta_j^*, \theta_1, \ldots, \theta_m \rangle$ has the value 1 for every eventual situation s. But the last formula of $\langle \beta_j^*, \theta_1, \ldots, \theta_m \rangle$ is the same as the last formula of $\langle \alpha^*, \alpha^*(X,Y), \theta_1, \ldots, \theta_m \rangle$, which implies that the latter is successful with respect to every eventual situation s. Thus, every realization of any instance α^* of α with f is successful with respect to every eventual situation, which, by definition, means that f accomplishes α.

Subcase 2.2: Assume $\Gamma(\alpha) = \bigcap y\delta$ and $\gamma = \delta(y/t)$, where either $t = x_\alpha$ or $t \in \text{Freeterms}(\alpha)$.

Then we define f as the function that, for every instnce

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

of α, acts as follows:

- $f\langle \alpha^* \rangle = X$, where $X = \langle \Gamma \ / \ \delta(y/b, x_1/a_1, \ldots, x_n/a_n) \rangle$, where

$$b = \begin{cases} t & \text{if } t \text{ is a constant} \\ a_i & \text{if } t \text{ is } x_i \text{ for one of the } 1 \leq i \leq n \\ \text{the constant } c_0 & \text{if } t = x_\alpha. \end{cases}$$

- For any realization $\langle \alpha^*, \alpha_1, \ldots, \alpha_m \rangle$ with $m \geq 1$ where α_1 is an instance of one of the β_j $(1 \leq j \leq k)$, f returns the same value as f_j does for the realization $\langle \alpha_1, \ldots, \alpha_m \rangle$. If here there are more than one such β_j, the one with the smallest j is selected.

Observe that X is indeed an action for α^* and $\alpha^*(X)$ is an instance of π. In particular, $\alpha^*(X) = \pi(y/b, x_1/a_1, \ldots, x_n/a_n)$.

We claim that f accomplishes α. The claim can be justified using the same argument as the one used in the Subcase 2.1.

Case 3: α follows from $\overline{\alpha}, \pi_1, \ldots, \pi_e$ by the R-rule.

Let $\beta_1, \ldots, \beta_k$ be all the proper quasireactions for α (so that $\pi_1, \ldots, \pi_e$ are among them). By Lemma 6.2, each of these formulas has a proof shorter than l and hence, by the induction hypothesis, there are action strategies $f_1, \ldots, f_k$ that accomplish $\beta_1, \ldots, \beta_k$, respectively.

Let then f be the function that, for every instnce

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

of α, acts as follows:

- $f\langle \alpha^* \rangle = \langle \rangle$;

- For any realization $\langle \alpha^*, \alpha_1, \ldots, \alpha_m \rangle$ with $m \geq 1$ where α_1 is an instance of one of the β_j $(1 \leq j \leq k)$, f returns the same value as f_j does for the realization $\langle \alpha_1, \ldots, \alpha_m \rangle$. If here there are more than one such β_j, the one with the smallest j is selected.

We claim that f accomplishes α.

Indeed, consider an arbitrary instance

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

of α, an arbitrary realization $\langle \alpha^*, \theta_1, \ldots, \theta_m \rangle$ of α^* and an arbitrary eventual situation s. We need to show that $\langle \alpha^*, \theta_1, \ldots, \theta_m \rangle$ is successful with respect to s.

There are two subcases to consider:

Subcase 3.1: Assume $m = 0$. Then α^* is the last formula of the realization and, by Lemma 3.5, the realization is successful with respect to s iff $s(\alpha^*) = 1$. So, it suffices to show that $s(\alpha^*) = 1$. But indeed, obviously $\overline{\alpha}(x_1/a_1, \ldots, x_n/a_n)$ is the primitivization of α^*. And since $\overline{\alpha}$ is provable in classical logic, so is $\overline{\alpha}(x_1/a_1, \ldots, x_n/a_n)$. Then, by Fact 3.3, we have $s\big(\overline{\alpha}(x_1/a_1, \ldots, x_n/a_n)\big) = 1$ whence, by Fact 3.2, $s(\alpha^*) = 1$.

Subcase 3.2: Assume now $m \geq 1$. Since $f\langle \alpha^* \rangle = \langle\rangle$, θ_1 must be a proper reaction for α^*. Therefore, by Lemma 6.3, θ_1 is an instance of a proper quasireaction for α. As $\beta_1, \ldots, \beta_m$ are all the proper quasireactions for α, θ_1 must be an instance of one of them. Let j be the smallest number ($1 \leq j \leq m$) such that θ_1 is an instance of β_j. Then it can be easily seen from the way f is defined that $\langle \theta_1, \ldots, \theta_m \rangle$ is a realization of θ_1 with the action strategy f_j. It was our assumption that the latter accomplishes β_j and hence θ_1. Therefore, by Lemma 3.5, $s(\theta_m) = 1$. And since θ_m is also the last formula of $\langle \alpha^*, \theta_1, \ldots, \theta_m \rangle$, this realization is also successful with respect to s. □

7 Stronger versions of the soundness theorem

As we remember, our assumption was that the set C of constants (and hence the universe of discourse as we assumed that the latter is nothing but C) is countably infinite. An analysis of the proof of Theorem 6.4 can reveal that the only assumption about C we employed was that c_0 is an element of C (Subcase 2.2). So that we have:

Fact 7.1 Theorem 6.4 remains valid for the case when the universe of discourse (set of constants) has any nonzero finite cardinality.

The following theorem is a constructive version of the soundness theorem for **L**:

Theorem 7.2 There is an effective procedure that takes as an input a proof of a formula α in **L** and returns an action strategy that accomplishes α.

PROOF A straightforward analysis of the proof of Theorem 6.4. □

The following theorem is even stronger:

Theorem 7.3 (The Universal Action Strategy Theorem) There is an action strategy ("the universal action strategy") that accomplishes every accomplishable task.

PROOF Here is how the universal action strategy works: Given a task α, it starts looking for a proof of α in **L**. By the soundness theorem for **L**, such a proof exists as long as α is accomplishable. And since **L** is semidecidable, there is a procedure that finds such a proof sooner or later when it exists. So the universal action strategy may employ this procedure and, if α is accomplishable, find an **L**-proof for it. Once a proof is found, the universal strategy employs the procedure from Theorem 7.2 and constructs an action strategy f that accomplishes α. After that, it uses f to accomplish α. □

8 The completeness of L

Let $x_1, \ldots, x_n$ be all the free variables of α. We say that an instance

$$\alpha(x_1/a_1, \ldots, x_n/a_n)$$

of α is *distinctive*, if each a_i is different from any constant occurring in α as well as from any a_j $(1 \leq j \leq n)$ with $j \neq i$.

Lemma 8.1 If β is an action (resp. proper action) for a distinctive instance of α, then β is a distinctive instance of a quasiaction (resp. proper quasiaction) for α.

PROOF Assume $x_1, \ldots, x_n$ are all the free variables of α,

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

is a distinctive instance of α and $\beta^* = \alpha^*(X)$ is an action for α^*.

We prove the lemma by induction on the length of X.

Basis: If $X = \langle\rangle$, i.e. the action X is not proper, then $\beta^* = \alpha^*$. As α is a (non-proper) quasiaction for itself and α^* is an instance of it, we are done.

Inductive step: Assume $X = \langle E_1, \ldots, E_k, \Gamma/\gamma^* \rangle$ $(k \geq 0)$. Then X is a proper action and $\alpha^*(X) = \theta^*(\Gamma/\gamma^*)$, where $\theta^* = \alpha^*(\langle E_1, \ldots, E_k \rangle)$. By the induction hypothesis, θ^* is a distinctive instance of a quasiaction θ for α. It suffices to show that $\theta^*(\Gamma/\gamma^*)$ is a distinctive instance of an elementary quasiaction for θ and hence of an proper quasiaction for α.

There are two cases to consider:

Case 1: Γ/γ^* is an elementary $\sqcap$-action for α^*, i.e. $\Gamma(\alpha^*) = \beta_0^* \sqcap \beta_1^*$ and γ^* is one of the β_i^*, $i \in \{0, 1\}$. Fix this i.

Since $\Gamma(\theta^*) = \beta_0^* \sqcap \beta_1^*$ and θ^* is a distinctive instance of θ, we must have $\Gamma(\theta) = \beta_0 \sqcap \beta_1$, where $\beta_0^* \sqcap \beta_1^*$ is a distinctive instance of $\beta_0 \sqcap \beta_1$. Then Γ/β_i is an

elementary quasiaction for θ, and obviously $\theta^*(\Gamma/\beta_i^*)$ is a distinctive instance of $\theta(\Gamma/\beta_i)$.

Case 2: Γ/γ^* is an elementary $\bigsqcap$-action for α^*, i.e. $\Gamma(\alpha^*) = \bigsqcap y\beta^*$ and γ^* is $\beta^*(y/b)$ for some constant b. Fix this constant.

Since $\Gamma(\theta^*) = \bigsqcap y\beta^*$ and θ^* is a distinctive instance of θ, we must have $\Gamma(\theta) = \bigsqcap y\beta$, where β^* is an instance of β. Let t be the term defined by

$$t = \begin{cases} b, \text{ if } b \text{ occurs in } \theta \\ x_i, \text{ if } b = a_i \text{ for one of the } i \text{ with } 1 \leq i \leq n \\ x_\theta \text{ otherwise} \end{cases}$$

Then $\Gamma/\beta(y/t)$ is an elementary quasiaction for θ, and obviously $\theta^*\big(\Gamma/\beta^*(y/b)\big)$ is a distinctive instance of $\theta\big(\Gamma/\beta(y/t)\big)$. □

Lemma 8.2 Assume f is an action strategy, α^* is a closed formula, $f\langle\alpha^*\rangle = X$, Y is a reaction for α^* and $\alpha^*(X, Y)$ is not accomplishable. Then f does not accomplish α^*.

PROOF Indeed, assume the conditions of the lemma and assume, for a contradiction, that f accomplishes α^*. Let f' be the action strategy such that for any realization $\langle\alpha_0, \ldots, \alpha_k\rangle$, $f'\langle\alpha_0, \ldots, \alpha_k\rangle = f\langle\alpha^*, \alpha_0, \ldots, \alpha_k\rangle$.

Since $\alpha^*(X, Y)$ is not accomplishable, there is a situation s and a realization

$$R = \langle\alpha^*(X, Y),\ \alpha_1,\ \ldots,\ \alpha_k\rangle$$

of $\alpha^*(X, Y)$ with strategy f' such that R is unsuccessful with respect to s. But notice that then

$$\langle\alpha^*, \alpha^*(X, Y), \alpha_1, \ldots, \alpha_k\rangle$$

is a realization of α^* with strategy f. By Lemma 3.5, the latter is also unsuccessful with respect to s because it has the same last formula as R does. □

Lemma 8.3 If $\mathbf{L} \nvdash \alpha$ and α^* is a distinctive instance of α, then α^* is not accomplishable.

PROOF Assume $\mathbf{L} \nvdash \alpha$ and α^* is a distinctive instance of α.

We are going to show that α^* is not accomplishable by induction on the additive complexity of α.

Basis: Assume α is primitive. $\mathbf{L} \nvdash \alpha$ implies that α is not provable in classical logic, and therefore α^* is not provable in classical logic, either (which generally would not necessarily be the case if α^* was not a distinctive instance of α). In view of Fact 3.3, this means that here is a situation s such that $s(\alpha^*) = 0$. As α^* is primitive, the only realization of it is $\langle\alpha^*\rangle$ and, by Lemma 3.5, this realization is not successful with respect to s. Thus, α^* is not accomplishable.

Inductive step: Assume α is not primitive. Let f be an arbitrary action strategy. We need to show that f does not accomplish α^*, i.e. there is a realization of α^* that is not successful with respect to some eventual situation.

There are two cases to consider:

Case 1: Assume $\overline{\alpha}$ is not provable in classical logic. It is easy to see that then $\overline{\alpha^*}$ is not provable in classical logic, either, because it is a distinctive instance of $\overline{\alpha}$. Therefore, by Fact 3.3 and Fact 3.2, there is an eventual situation s such that

$$s(\alpha^*) = 0. \tag{2}$$

If $f\langle\alpha^*\rangle = \langle\rangle$, then $\langle\alpha^*\rangle$ is a realization of α^*. It follows from (2) by Lemma 3.5 that this realization is not successful with respect to s.

Assume now $f\langle\alpha^*\rangle = X$ for some nonempty action X for α^*. Then, according to Lemma 8.1, there is a proper quasiaction π for α such that $\alpha^*(X)$ is a distinctive instance of π. $\mathbf{L} \nvdash \pi$ because otherwise, by the A-rule, we would have $\mathbf{L} \vdash \alpha$. Also, the additive complexity of π lower than that of α. Therefore, by the induction hypothesis, no distinctive instance of π is accomplishable. In particular, $\alpha^*(X)$ is not accomplishable. Then, by Lemma 8.2 (with $Y = \langle\rangle$), f does not accomplish α^*.

Case 2: Assume now $\overline{\alpha}$ is provable in classical logic. Assume $x_1, \ldots, x_n$ are all the distinct free variables of α, so that

$$\alpha^* = \alpha(x_1/a_1, \ldots, x_n/a_n)$$

for some distinct constants $a_1, \ldots, a_n$.

Since α is not provable in $\mathbf{L}$, there is an elementary quasireaction

$$\pi = \alpha(\Gamma/\gamma)$$

for α that is not provable (otherwise α would be derivable by the R-rule). There are two subcases to consider:

Subcase 2.1: Assume $\Gamma(\alpha) = \xi_0 \sqcap \xi_1$ and $\gamma = \xi_i$ for one of the $i \in \{0, 1\}$. Fix this i.

Let Y be the reaction $\langle \Gamma \ / \ \xi_i(x_1/a_1, \ldots, x_n/a_n)\rangle$. Obviously $\alpha^*(Y)$ is a distinctive instance of π. Hence, by Lemma 8.1, every action for $\alpha^*(Y)$ is a distinctive instance of a quasiaction for π. Clearly a quasiaction for π can not be provable, for otherwise, by the A-rule, π would be provable, too. Therefore, by the induction hypothesis, we have:

$$\text{No action for } \alpha^*(Y) \text{ is accomplishable.} \tag{3}$$

Consider an arbitrary action strategy f. Let $f\langle\alpha^*\rangle = X$. In view of Fact 2.4, X would also be an action for $\alpha^*(Y)$. Therefore, by (3), $\alpha^*(X, Y)$ is not accomplishable. Then, by Lemma 8.2, f does not accomplish α^*.

Subcase 2.2: Assume $\Gamma(\alpha) = \bigsqcap y\xi$ and $\gamma = \xi(y/x_\alpha)$.

Let b be a constant that does not occur in α^*, and let Y be the reaction $\langle \Gamma/\delta(y/b, x_1/a_1, \ldots, x_n/a_n)\rangle$. Observe that $\alpha^*(Y)$ is a distinctive instance of π. Further we can repeat the argument of the Subcase 2.1 to show that f does not accomplish α^*. $\Box$

The following theorem is an immediate consequence of Lemma 8.3:

Theorem 8.4 (Completeness of L) If α is accomplishable, then $\mathbf{L} \vdash \alpha$.

Remark 8.5 As we remember, the definition of action strategy assumes that this strategy is effective. However, in the proof of Theorem 8.4, this assumption was never used. Thus, **L** remains complete even if the requirement of effectiveness of action strategies is removed and any strategies are allowed.

9 Beyond finite tasks

The basic connectives of the language of **L** do not form a complete set of operators in any natural sense of expressive power, and **L** is open to interesting extensions by means of adding new operators to its language.

The language defined in Section 2 only allows us to express *finite tasks* — tasks that always have finite-length realizations, even though the number of different realizations of a given task can be infinite. This sort of tasks are sufficient for modelling situations with bounded physical resources, such as a fixed number n of transcontinental missiles each of which can be used only once, or n pieces of litmus paper or disposable pregnancy test devices. When $n > 1$, such resources can be expressed with $\wedge$-conjunctions of n identical conjuncts.

However, even though in real life all resources are limited and no task with infinitely long realizations can really be accomplished, assuming the existence of inexhaustible resources, such as an unlimited number of missiles or a reusable version of litmus paper, is often a convenient abstraction. The need in having means to express reusable resources is especially great when we deal with purely informational tasks/resources. For example, a computer program that calculates square roots carries out the task

$$\sqcap x \sqcup y(y = \sqrt{x}),$$

but it does so an unlimited number of times rather than once. So that the task carried out by such a program should be expresses by the infinite conjunction

$$\sqcap x \sqcup y(y = \sqrt{x}) \ \wedge \ \sqcap x \sqcup y(y = \sqrt{x}) \ \wedge \ \sqcap x \sqcup y(y = \sqrt{x}) \ \wedge \ \ldots$$

Obviously this task can produce infinitely long realizations — say, the master can request finding the square root for 1, then for 2, then for 3, ... Such a realization should be considered successful, if its (finite) projection on each of the conjuncts is successful.

Of course we do not want to have infinitely long formulas in the language. We can use the finite expression

$$\uparrow\alpha$$

to stand for $\alpha\wedge\alpha\wedge\ldots$. Applying an action or a reaction (or a combination of those two) to the (sub)formula $\uparrow\alpha$ will result in replacing $\uparrow\alpha$ by $(\uparrow\alpha)\wedge\alpha'$, where α' is the result of applying such an action/reaction to α in the sense of Section 2. This way, in the process of realization, the unchanging part $\uparrow\alpha$ can produce more and more new "activated" conjuncts, but the number of those conjuncts will remain finite at any particular stage, so that every formula in the realization will be a finite expression.

This semantics makes the operator $\uparrow$ a counterpart of the exponential operator ! of linear logic.

In Section 3 we mentioned another operator, $\bigwedge$, that also takes us beyond finite tasks and that is, in fact, more expressive than $\uparrow$. As we remember, the meaning of

$$\bigwedge x\alpha(x)$$

is

$$\alpha(c_0)\wedge\alpha(c_1)\wedge\ \ldots$$

In the Introduction we noted that the classical operators of our language correspond to the multiplicative operators of linear logic. This statement, however, should be restricted to propositional operators only. The classical quantifier $\forall$ has no counterpart in linear logic. What can be called a multiplicative quantifier is not $\forall$ but $\bigwedge$, for it is the same to the multiplicative conjunction $\wedge$ as $\bigsqcap$ is to the additive conjunction $\sqcap$.

Just as this is the case with $\uparrow\alpha$, at every stage of a realization, only a finite number of conjuncts hidden in $\bigwedge x\alpha(x)$ will be activated (be modified by proper actions/reactions). However, as all these conjuncts are different, we need to keep track of exactly which of them have been activated. For this reason, the syntax for the operator $\bigwedge$ should be

$$\bigwedge x\notin\{a_1,\ldots,a_n\}\alpha(x)$$

(where $a_1,\ldots,a_n$ are constants) rather than just $\bigwedge x\alpha(x)$. The meaning of this expression is the conjunction of all the formulas $\alpha(a)$ except those with $a\in\{a_1,\ldots,a_n\}$. The expression $\bigwedge x\alpha(x)$ could then be understood as an abbreviation for $\bigwedge x\notin\{\}\alpha(x)$.

Applying an action or a reaction (or a combination of those two) to the hidden conjunct $\alpha(b)$, where $b\notin\{a_1,\ldots,\alpha_n\}$, will result in replacing

$$\bigwedge x\notin\{a_1,\ldots,a_n\}\alpha(x)$$

by

$$\left(\bigwedge x\notin\{b,a_1,\ldots,a_n\}\alpha(x)\right)\wedge\alpha'(b),$$

where $\alpha'(b)$ is the result of applying that action/reaction to $\alpha(b)$ in the sense of Section 2.

Note that as long as x does not occur in α and the set of constants is infinite, $\uparrow\alpha$ is equivalent to $\bigwedge x\alpha$, so that there is no special need in having $\uparrow$ as a basic operator together with $\bigwedge$.

Problem: Is the (sound and complete) logic of tasks semidecidable when $\bigwedge$ is added to its language?

References

[1] S.Abramsky and R.Jagadeesan, *Games and full completeness for multiplicative linear logic.* Journal of Symbolic Logic 59 (1994), no. 2, pp.543-574.

[2] A.Blass, *A game semantics for linear logic.* Annals of Pure and Applied Logic, v.56 (1992), pp. 183-220.

[3] W.Felscher, *Dialogues, strategies and intuitionistic provability.* Annals of Pure and Applied Logic, v.28 (1985), no.3, pp.217-254.

[4] J.Y.Girard, *Linear logic.* Theoretical Computer Science, v.50-1 (1987), pp. 1-102.

[5] A.Haas, *A syntactic theory of knowledge and action.* Artificial Intelligence, vol.28, no.3, 1986, pp.245-292.

[6] *Game-Theoretical Semantics: Essays on Semantics by Hintikka, Carlson, Peacocke, Rantala and Saarinen* (edited by E.Saarinen). Dordrecht, Holland, 1979.

[7] G.Japaridze, *A simple proof of arithmetical completeness for* Π_1*-conservativity logic.* Notre Dame Journal of Formal Logic 35 (1994), No 3. pp.346-354.

[8] G.Japaridze, *A constructive game semantics for the language of linear logic.* Annals of Pure and Applied Logic, v.85 (1997), pp. 87-156.

[9] G.Japaridze, *The Logic of Resources and Tasks.* Ph.D. Thesis. University of Pennsylvania, Philadelphia, 1998. 145 pages.

[10] G.Japaridze, *A decidable first order epistemic logic.* Proceedings of the Georgian Academy of Sciences N 1/2, 2000, pp.81-95.

[11] G.Japaridze, *A task semantics for the language of linear logic.* Bulletin of the Georgian Academy of Sciences, v. 163 (2001), N 1, pp.5-7.

[12] K.Lorenz, *Dialogspiele als semantische Grundlage von Logikkalkülen.* Arch. Math. Logik Grundlag., v.11 (1968), pp.32-55, 73-100.

[13] P.Lorenzen, *Ein dialogisches Konstruktivitätskriterium.* In: Infinitistic Methods (PWN, Warsaw, 1959), pp.193-200.

[14] J.McCarthy and P.Hayes, *Some philosophical problems from the standpoint of Artificial Intelligence.* In: B.Meltzer, ed., Machine Intelligence 4, 1969, pp.463-502.

[15] R.Moore, *A formal theory of knowledge and action.* In: Hobbs and Moore, eds., 1985.

[16] S.Russel and P.Norwig, *Artificial Intelligence: A Modern Approach.* Prentice-Hall, 1995.