Ivan M. Havel

The concept of indirectness in artificial intelligence

Kybernetika, Vol. 8 (1972), No. 2, (154)--164

Persistent URL: http://dml.cz/dmlcz/124936

KYBERNETIKA — VOLUME 8 (1972), NUMBER 2

The Concept of Indirectness in Artificial Intelligence

IVAN M. HAVEL

The concept of indirectness (using of tools) in automatic problem solving is investigated, and a formal approach is suggested within the framework of the Situation Calculus. The "Robot and Box" problem serves as an illustrative example throughout the paper.

1. INTRODUCTION

Typical activity in the field of Artificial Intelligence consists in writing computer programs and/or constructing sophisticated devices performing various particular tasks usually demanding human intelligence and reasoning. In 1969 a robot, built by the Artificial Intelligence Group at Stanford Research Institute, succeeded in performing a relatively complicated task requiring an auxiliary action which may be without exaggeration qualified as "using a tool". (This task, known as the "Robot and Box" problem, will be described later.) The class of problems characterized by some amount of indirectness appears to be of great importance in formulating and understanding the principles of artificial reasoning.

Informally, by *indirectness* in a solution of some problem we mean the requirement for an auxiliary device, tool, or action not obviously needed at the start of the problem. In a natural way we will talk about *indirectness of the problem itself* in the case when every solution of the problem appears to be indirect. Even if such a concept of indirectness is intuitively quite clear and obvious, there is no known straightforward way of characterizing it in a more formal manner. An attempt is made in this paper to suggest a formalization of this concept in the framework of the first-order state-space problem-solving method sometimes called the Situation Calculus ([3], [1]).

Our investigation leads us to conclude that while the concept of indirectness admits formalization in principle, its particular form (e.g., the position of a borderline region between the direct and the indirect) is dependent not only on the environment and the problem itself, but also on the observer's evaluation of the situation. (The

author believes that such a conclusion is in agreement with our "nonartificial" concept of indirect behavior, which is in every particular case influenced by long-time individual experience and by a degree of familiarity with our environment.)

2. THE SITUATION CALCULUS

We will present a very brief outline of the Situation Calculus and the way it is used in problem solving. The unsatisfied reader is referred to the following sources: [4] for general introductory material; [4], Ch. 7 and [2] for application of the predicate calculus in state-space problem solving; [3] for the concept of the Situation Calculus and general ideas behind it; and [1], Appendix I, for a brief description of the Situation Calculus in the form we use it here.

For better understanding and illustration we use as an example the "Robot and Box" problem [1]. Our formulation of the corresponding Situation Calculus is slightly different from that presented in [1], and both of them are probably much simpler than the formulation needed in the real experiment. The "Robot and Box" problem is an analogical adaptation of the well-known "Monkey and Banana" problem (cf. [4]). It is formulated as follows:

The robot is in a room in which a box is resting on the top of a platform (Fig. 1(a)). Tke robot's problem is to push the box off the platform and onto the floor. The robot is on wheels and cannot reach the box directly. However, there is a ramp in the room, which the robot can move to a location adjoining the platform and then roll up the ramp onto the platform and finish the task (Fig. 1(b), (c), (d)).

Such a problem can be formalized using the state-space approach: every situation (mutual position of all objects) characterizes a *state*. The task can be expressed as a search for a transformation from some *initial state* s_0 (e.g., Fig. 1(a)) to a *final state* s (e.g., Fig. 1(d)) using a sequence of *state-transforming functions* (actions from the robot's repertoire). In the Situation Calculus various states are characterized by truth values of certain *predicates*. Properties of these predicates as well as properties of the state-transforming functions are expressed by means of a set of *axioms*. A solution can then be obtained by application of standard theorem-proving techniques.

Formally* the Situation Calculus (SC) is a sixtuple

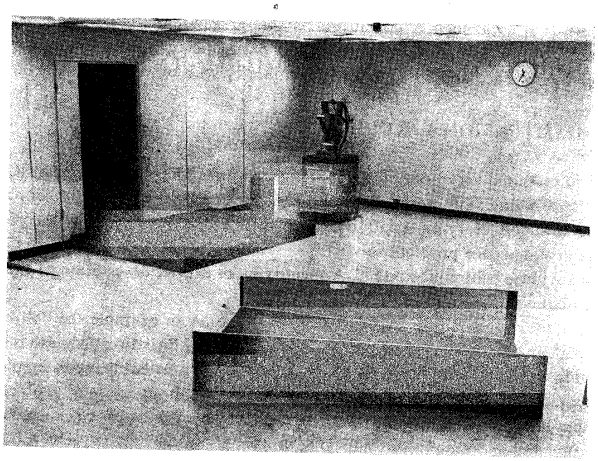
$$\mathscr{S} = \langle \mathscr{A}, \mathscr{P}, \mathscr{F}, S, s_0, \Omega \rangle$$

where Ω is the *universe of discourse*; in our example it is the set of objects

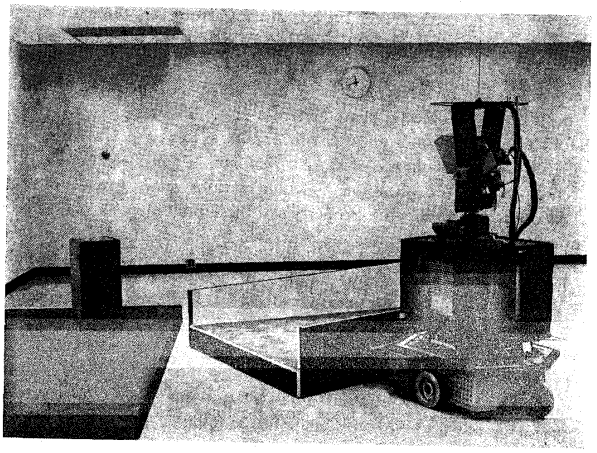
$$\Omega = \{\text{robot, floor, box, platform, ramp}\} .$$

In general, objects may be of various kinds (e.g., numerical coordinates).

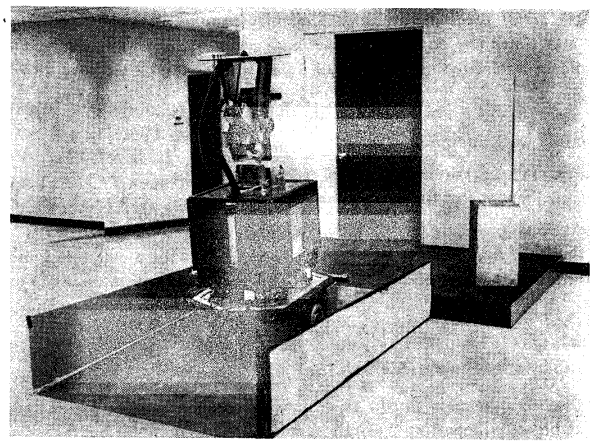
* A completely formal presentation would require some additional constructions (e.g. distinguishing the syntax and semantics of $\mathscr{S}$).



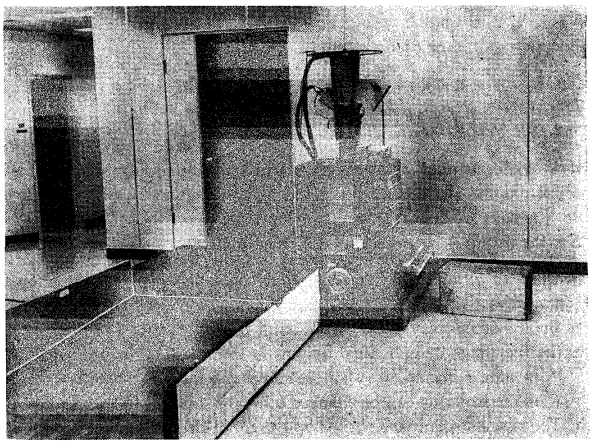
(a)



(b)



(c)



(d)

Fig. 1. The robot and the box. The sequence from (a) to (d) shows the robot using the ramp as a tool to climb up on the platform and push the box off the platform. (Photo courtesy S. L. Coles, SRI.)

$\mathscr{P}$ is the *set of predicate symbols*. The corresponding predicates are defined for objects and states (by convention, the last argument is always reserved for the state) and characterize the properties of states. In our example we have

$$\mathscr{P} = \{\text{ON, AT}\}$$

with the following interpretation:

ON(x, y, s) means that object x is *on* object y in state s;
AT(x, y, s) means that object x is *at* a location adjoining object y in state s.

An additional predicate '=' means equality of objects in the usual way.

$\mathscr{F}$ is the *set of function symbols*. Functions are defined for some objects and states (again the last argument stands for state) with values in the set of states. In our example we have

$$\mathscr{F} = \{\text{GOTO, PUSH, MOVE, ROLLUP}\}$$

with the following interpretation:

GOTO(x, y, s) maps s into a new state in which the object x (the agent) has arrived at object y (departing 'in' state s; cf. Ax. 10);

PUSH(x, y, s) maps s into a new state in which agent x has pushed object y (cf. Ax. 11);

MOVE(x, y, z, s) maps s into a new state in which agent x has moved object y to a location adjoining object z (cf. Ax. 12);

ROLLUP(x, y, s) maps s into a new state in which agent x has rolled up object y (Ax. 13).

S is the *set of states*. It can be understood as a collection of various mutual configurations of objects in Ω in terms of predicates corresponding to symbols of $\mathscr{P}$ and obeying the axioms. (In our example, predicates are binary over Ω and $|\Omega| = 5$, therefore S always has less than $2^{5^2} \cdot 2^{5^2} = 2^{50}$ states.)

$s_0 \in S$ is the *initial state* which satisfies some initial axioms (cf. Ax. 1–4).

$\mathscr{A}$ is the *set of axioms*. Every axiom is a first-order formula containing, besides logical connectives, variables ranging over Ω and S and constants from Ω, $\mathscr{F}$, $\mathscr{P}$. Below is the list of axioms for the "Robot and Box" problem. There are four groups of them; the meanings of individual axioms is intuitively clear (for instance, Ax. 12 means: "If, in state s, the robot is 'at' the ramp and the ramp is 'at' some another object x, then the situation can be changed to a new state s' in which the robot is 'on' object x. The new state can be obtained by the robot's rolling up the ramp.")

Axioms for initial state:

Ax. 1 ON(robot, floor, s_0)
Ax. 2 ON(box, platform, s_0)

Ax. 3 ON(platform, floor, s_0)
Ax. 4 ON(ramp, floor, s_0)

General properties of predicates ON *and* AT:

Ax. 5 $(\forall x, s)\,[\neg \mathrm{ON}(x, x, s)]$
Ax. 6 $(\forall x, y, z, s)\,[\mathrm{ON}(x, y, s) \,\&\, \mathrm{ON}(x, z, s) \Rightarrow y = z]$
Ax. 7 $(\forall x, s)\,[\mathrm{AT}(x, x, s)]$
Ax. 8 $(\forall x, y, s)\,[\mathrm{AT}(x, y, s) \Rightarrow \mathrm{AT}(y, x, s)]$
Ax. 9 $(\forall x, y, z, s)\,[\mathrm{AT}(x, y, s) \Rightarrow [\mathrm{ON}(x, z, s) \Leftrightarrow \mathrm{ON}(y, z, s)]]$

(Axioms 6 and 8 are included for completeness, they are not necessary for the solution of the problem.)

Properties of robot's actions:

Ax. 10 $(\forall x, y, s)\,[\mathrm{ON}(\mathrm{robot}, x, s) \,\&\, \mathrm{ON}(y, x, s) \Rightarrow \mathrm{AT}(\mathrm{robot}, y, \mathrm{GOTO}(\mathrm{robot}, y, s))]$
Ax. 11 $(\forall x, y, z, s)\,[\mathrm{AT}(\mathrm{robot}, x, s) \,\&\, \mathrm{ON}(x, y, s) \,\&\, \mathrm{ON}(y, z, s) \Rightarrow \mathrm{ON}(x, z, \mathrm{PUSH}(\mathrm{robot}, x, s))]$
Ax. 12 $(\forall x, s)\,[\mathrm{AT}(\mathrm{robot}, \mathrm{ramp}, s) \,\&\, \mathrm{AT}(\mathrm{ramp}, x, s) \,\&\, \neg(x = \mathrm{robot}) \Rightarrow \Rightarrow \mathrm{ON}(\mathrm{robot}, x, \mathrm{ROLLUP}(\mathrm{robot}, \mathrm{ramp}, s))]$
Ax. 13 $(\forall x, y, z, s)\,[\mathrm{ON}(y, z, s) \,\&\, \mathrm{ON}(x, z, s) \,\&\, \mathrm{AT}(\mathrm{robot}, x, s) \Rightarrow \Rightarrow \mathrm{AT}(x, y, \mathrm{MOVE}(\mathrm{robot}, x, y, s))]$

Frame axioms:

Ax. 14 $(\forall x, y, z, s)\,[\mathrm{ON}(x, y, s) \Rightarrow \mathrm{ON}(x, y, \mathrm{GOTO}(\mathrm{robot}, z, s))]$
Ax. 15 $(\forall x, y, z, s)\,[\mathrm{ON}(x, y, s) \Rightarrow \mathrm{ON}(x, y, \mathrm{MOVE}(\mathrm{robot}, x, z, s))]$
Ax. 16 $(\forall x, y, s)\,[\mathrm{AT}(\mathrm{robot}, x) \Rightarrow \mathrm{AT}(\mathrm{robot}, x, \mathrm{MOVE}(\mathrm{robot}, x, y, s))]$
Ax. 17 $(\forall x, y, s)\,[\mathrm{ON}(x, y, s) \,\&\, \neg(x = \mathrm{robot}) \Rightarrow \mathrm{ON}(x, y, \mathrm{ROLLUP}(\mathrm{robot}, \mathrm{ramp}, s))]$

(Frame axioms express what does *not* get changed by the functions. We include only those which are necessary for the solution.)

Now the realizability of the task

"Push the box that is on the platform onto the floor"

can be expressed by the existential formula

(*) $\qquad (\exists s)\,[\mathrm{ON}(\mathrm{box}, \mathrm{floor}, s)]\,.$

To represent the task itself, we use the following notation

(**) $\qquad (!\, s)\,[\mathrm{ON}(\mathrm{box}, \mathrm{floor}, s)]$

(as a matter of fact the above verbal formulation is more accurately represented by the following formula:

$$(!\, s)\,[\mathrm{ON}(\mathrm{box}, \mathrm{floor}, \mathrm{PUSH}(\mathrm{robot}, \mathrm{box}, s))])\,.$$

It is not our purpose to describe how the solution is obtained. Let us only point out that formula (*) is fed as an input (together with the axioms) to a theorem-proving program extended by an answer-extraction routine. Such a program not only proves formula (*) – in the case it is provable, i. e., when s is 'reachable' from s_0 – but also outputs a description of the final state s in terms of state-transforming functions. For our robot the answer would be

(***) s = PUSH(robot, box, GOTO(robot, box, ROLLUP(robot, ramp, MOVE(robot, ramp, platform, GOTO(robot, ramp, s_0))))) .

Note that our example suffers from a lack of generality in some respects. For instance, there is only one active agent (the robot), the universe consists of a finite and fixed number of objects, the predicates are everywhere defined, and the solution is off-line in the sense that the final answer (***) must be determined before it can be executed by the robot (in fact the functions are generated by the theorem prover in reverse order). Since our aim is to investigate the concept of indirectness and not to create a general and comprehensive theory, we are not overly constrained by the above restrictions.

3. THE CONCEPT OF INDIRECT TASKS

Everyone would probably agree that the above example of the "Robot and Box" problem to some extent contains an indirect element. The statement of the problem (either verbal or formal) and the most relevant environment would appear identical even if there were no ramp available. However, in that case the problem would *not* have a solution. Therefore, a first suggestion might be: qualify as indirect a task requiring, in any answer, some object or action that is not mentioned in the formulation (cf. the informal definition of indirectness in introduction). In particular, there is no doubt that objects like the box or robot itself (the robot can be thought of as implicitly included in its task) should not be considered as tools.* However, the above suggestion does not appear completely adequate; consider the following task for our robot:

"Move the ramp to the platform"

or, formally,

$$(!\, s)\,[\mathrm{AT}(\mathrm{ramp}, \mathrm{platform}, s)] .$$

We do not expect any indirectness in this task; however, the answer to it would be

$$s = \mathrm{MOVE}(\mathrm{robot}, \mathrm{ramp}, \mathrm{platform}, \mathrm{GOTO}(\mathrm{robot}, \mathrm{ramp}, s_0))$$

where the (otherwise quite obvious) action 'GOTO' would qualify itself as an indirect

* We do not wish to get into the question of whether the robot is a tool of its computer or of the experimentor himself.

one. We can imagine another point of view, indeed, where the 'GOTO' action would be highly indirect. We come to the following observation: In whatever way the indirectness is defined, it should include some additional 'parameter' which would make the borderline between 'direct' and 'indirect' to some extent adjustable.

Now, if the ramp in our example is qualified as an indirect object, what about the action 'ROLLUP' which has no use without the ramp. It seems that some objects and/or actions are closely related in the sense that the directness or indirectness of one implies the same property for the other. This is our second observation.

Finally, we observe that indirectness can be sometimes associated with *objects* (elements of Ω) and sometimes with *functions* (elements of $\mathscr{F}$). Given a definition of indirectness, we will call indirect objects *tools* and indirect functions *tricks*. Thus, walking around obstacles may be called a trick (with respect an appropriate definition of indirectness). In our example the function ROLLUP is a trick with the ramp as a tool. In our following formulation tools can be uderstood as a special case of tricks in the same way and for the same reason as individual constants are special cases of functions (namely 0-ary ones).

Now we will formalize our concept of indirectness in the framework of the SC. Let

$$\mathscr{S} = \langle \mathscr{A}, \mathscr{P}, \mathscr{F}, S, s_0, \Omega \rangle$$

and

$$\mathscr{S}' = \langle \mathscr{A}', \mathscr{P}', \mathscr{F}', S', s_0', \Omega' \rangle$$

be two SC's. We shall call $\mathscr{S}'$ a *subcalculus* of $\mathscr{S}$ ($\mathscr{S}' \leqq \mathscr{S}$) iff the following holds:

(1) $\Omega' \subseteq \Omega$;

(2) $\mathscr{F}' \subseteq \mathscr{F}$;

(3) $\mathscr{P}' = \mathscr{P}$;

(4) Let $\mathscr{A}_0$ be a set of axioms obtained from $\mathscr{A}$ by

(i) replacing symbol s_0 by symbol s_0';
(ii) replacing all subformulas of the form '$x = a$' where x is a variable and $a \in \Omega - \Omega'$, by '*false*'.

Then $\mathscr{A}'$ contains exactly those axioms of $\mathscr{A}_0$ in which there is no occurrence of symbols from $\Omega - \Omega'$ or from $\mathscr{F} - \mathscr{F}'$.

(5) $S' = S/\approx$ (partition of S under $\approx$)

where $\approx$ is an equivalence relation defined as follows:

$s_1 \approx s_2$ iff for every n, every n-ary predicate P in $\mathscr{S}$ and every $x_1, \ldots, x_{n-1} \in \Omega'$ we have

$$P(x_1, \ldots, x_{n-1}, s_1) \Leftrightarrow P(x_1, \ldots, x_{n-1}, s_2) .$$

Intuitivelly, $s_1 \approx s_2$ when s_1 and s_2 are indistinquishable when 'observing' objects only from Ω'. In other words, S' is obtained from S by 'removing', or better to say 'losing interest in' objects from $\Omega - \Omega'$.

(6) $$s_0' = [s_0]_{\approx} \text{ (the equivalence class of } s_0).$$

The careful reader can easily fill in various formal details, e.g., the properties of functions and predicates corresponding to symbols in $\mathscr{F}'$ and $\mathscr{P}'$.

Consider our example of the SC $\mathscr{S} = \langle \mathscr{A}, \mathscr{P}, \mathscr{F}, S, s_0, \Omega \rangle$ for the "Robot and Box" problem. Let $\mathscr{S} = \langle \mathscr{A}', \mathscr{P}', \mathscr{F}', S', s_0, \Omega' \rangle$ be another SC where

$$\Omega' = \Omega - \{\text{ramp}\} = \{\text{robot, floor, box, platform}\},$$

$$\mathscr{F}' = \mathscr{F} - \{\text{ROLLUP}\} = \{\text{GOTO, MOVE, PUSH}\},$$

and

$$\mathscr{P}' = \mathscr{P}.$$

Furthermore let $\mathscr{A}'$ consist of Ax. 1–3, 5–11, and 13–16 and let S' be obtained from S by making equivalent all the states of S that differ only in a position of the ramp.

It can be easily verified that then $\mathscr{S}' \leqq \mathscr{S}$.

Let us now return to our main problem, the concept of indirectness. Let $\mathscr{S} = \langle \mathscr{A}, \mathscr{P}, \mathscr{F}, S, s_0, \Omega \rangle$ be a SC and let $\mathscr{S}' = \langle \mathscr{A}', \mathscr{P}', \mathscr{F}', S', s_0', \Omega' \rangle$ be its subcalculus such that $\Omega' \neq \Omega$ or $\mathscr{F}' \neq \mathscr{F}$ (in such a case we say that $\mathscr{S}'$ is a *proper subcalculus* of $\mathscr{S}$, $\mathscr{S}' < \mathscr{S}$).

Let

$$(!\, s)\, \mathscr{B}$$

be a task in $\mathscr{S}$ such that all individual objects and functions occurring in $\mathscr{B}$ are contained in Ω' and $\mathscr{F}'$ resp. Then $\mathscr{B}$ is also a formula in $\mathscr{S}'$. *We say that* $(!\, s)\, \mathscr{B}$ *is an indirect task in* $\mathscr{S}$ *with respect to* $\mathscr{S}'$ *iff the formula* $(\exists s)\, \mathscr{B}$ *is provable in* $\mathscr{S}$ *but not in* $\mathscr{S}'$. In accordance with our previous comments we call $\Omega - \Omega'$ the *set of tools in* $\mathscr{S}$ *with resepct to* $\mathscr{S}'$ and $\mathscr{F} - \mathscr{F}'$ the *set of tricks in* $\mathscr{S}$ *with respect to* $\mathscr{S}'$.

Thus for instance the task (**) is indirect with respect to the subcalculus of the "Robot and Box" problem as defined above. For, while (*) is provable in $\mathscr{S}$ (cf. (***)), it is not provable in $\mathscr{S}'$; the robot needs some trick or tool to get onto the platform. Note that in our formulation there is no subcalculus of the "Robot and Box" problem which would contain the ROLLUP function but not the ramp.

Selection of an appropriate subcalculus can, in general, be accomplished in many different ways (though not as many as one would think at the first sight, since 'removing' of one thing may force us to remove some other things too – as we have just seen). This ambiquity makes the concept of indirectness strongly dependent on the observer's choice.

4. CONCLUDING REMARKS

Once we have a more or less adequate concept of indirectness, we can try to investigate tasks with multiple level of indirectness. To show that such a generalization might be of some interest let us quote a passage from [1]:

"... any computer system capable of solving this class of problems characterized by one level of indirectness, which also contains a logically complete deductive component, could in principle handle problems having an arbitrary number of levels of indirectness required for their solution, subject to the constraints of computer memory and response time. It should be noted in passing that problems possessing merely a half-dozen levels of indirectness more than challenge human inqenuity."

This passage is, a however, a very informal comment and cannot be verified without an appropriate formalization of the concept of level of indirectness.

One approach to this problem within our formalization might be to specify *series of subcalculi* of the form

$$\mathscr{S}_0 < \mathscr{S}_1 < \ldots < \mathscr{S}_n = \mathscr{S} .$$

Unfortunately, not every such series would be intuitively acceptable as a formalization of the level-structure of problems, since it might not express mutual dependences or independences of various indirect actions. For example we would not consider the use of a tool A and later *for a different purpose* another tool B, as indirectness of level two. On the other hand, using some tool A *in order to acquire another tool* B would be a nice example of two-level indirectness. Both cases could possibly yield the same series.

Apparently the right way would be to introduce the concept of *subtasks* and to study their (in)directness. For instance

$$(!\, s)\,[\mathrm{AT}(\mathrm{ramp}, \mathrm{platform}, s)]$$

would be considered as a *direct* subtask of our *indirect* task

$$(!\, s)\,[\mathrm{ON}(\mathrm{box}, \mathrm{floor}, s)] .$$

Since solution of the subtask removes the indirectness of the task, it is concievable that if the subtask were indirect as well (e.g., the ramp would have to be assembled from pieces using various additional tools), we would be justified in regarding the complete task as indirect of higher level.

Acknowledgement. The author is grateful to Dr. L. Stephen Coles for careful reading and detailed comments on the manuscript. The paper was written during the author's stay at the University of California at Berkeley (Department of Computer Science).

(Received October 27, 1971.)

[1] L. S. Coles: An experiment in Robot Tool Using. SRI Artificial Intelligence Group, Tech. Note 41, Oct. 1970.
[2] C. C. Green: Application of Theorem Proving to Problem Solving. Proc. 1st Int. Joint Conf. on Artificial Intelligence, May 1969.
[3] J. McCarthy and P. Hayes: Some Philosophical Problems from the Standpoint of Artificial Intelligence. In: Machine Intelligence 4, 1969.
[4] N. J. Nilsson: Problem-Solving Methods in Artifical Intelligence. McGraw-Hill, 1971.

VÝTAH

Pojem nepřímé akce v umělé inteligenci

IVAN M. HAVEL

Práce se zabývá pojmem „nepřímé akce" – např. použití nástroje či jiné zprostředkované chování – při mechanickém řešení problémů. Je navržen formální přístup k definici tohoto pojmu v rámci tzv. situačního kalkulu. K přiblížení výkladu je použit jako příklad experiment s robotem a krabicí, který byl úspěšně realizován ve Stanfordském výzkumném ústavu v USA.

Ing. Ivan M. Havel, Ústav pro využití výpočetní techniky v řízení (Inst. Appl. Comp. Tech.), Loretánské nám. 3, Praha 1.