공학 연구논문집 제26권 제2호 pp. 157~164, 울산대학교
Journal of Engineering Research Vol. 26. No. 2, pp. 157~164 1995.
University of Ulsan.

Conceptual Structure of Data

Jae-Hak Bae
Department of Computer Science

〈Abstract〉

In this paper, we present a system of primitive concepts for expressing conceptualizations underlying the structure of a data and have outlined a method for planning on a data which utilizes such conceptualizations. Although a complete planning system based on the approach presented here has not been developed, we believe that this is a new conceptual approach to automatic programming.

개념적 자료 구조*

배 재 학
전자계산학과

〈요 약〉

본 논문에서는 자료에 내재한 구조 개념을 기술하는데 필요한 근원적인 표현 어휘 체제와, 이를 이용하여 표현된 자료를 대상으로 자료 처리 계획을 수립하는 방법에 대해 소개한다. 이러한 접근 방식에 기초한 계획 시스템은 현재 개발 중에 있지만, 이 시도는 자동 프로그래밍을 위한 새로운 개념적 해법의 모색이라고 생각한다.

* 본 연구는 1995 학년도 울산대학교 학술연구비 지원을 받았음

1. Introduction

There are already following evidences which support the fact that the behavior of a system can be derived from its structure. It is known that we can derive the structure of a program from the structure of its data. This is the key idea of the data structured program design methodology [2,4,7]. The same notion is claimed in a database design technique as the behavior design follows the structural design closely [1]. Likewise, it is assumed in qualitative reasoning about physical systems that a qualitative description of the behavior of a physical system can be derived from a qualitative description of its structure [3].

The ultimate goal of this research is to develop a theory behind these structure-behavior evidences. In the meantime, we study both how and why the operation sequence on a data can be derived from its structure, which achieves a goal operation on it. To do so, it was necessary to have a theory not only to represent them but also to deduce relationships among them. A theory, which is called Conceptual Dependency(CD) theory, has been developed both as a theory of language and as a meaning representation language that enhances the ability to build natural language understanding programs [5]. Moreover, scripts, plans, goals, and themes were introduced to augment CD as additional methods of representing the contextual information for story understanding and generation [6]. In addition, a theory of plans, which is based on CD, is developed and incorporated in a plan generation program called PANDORA(Plan ANalysis with Dynamic Organization, Revision, and Application) [8]. In these respects, CD seems to be the theory which meets most of our needs.

We, however, want a theory of plans to account for the interaction between a goal operation on a data and the structure of the data. As far as we know, there is no such theory of plans, and the theory behind PANDORA is not the one as well. To study with CD both how and why the operation plan on a data, which achieves a goal operation on it, can be derived from its structure, we first attempted to represent its structure in CD. This paper presents our conceptual approach to study the structure of a data. The structure of a data which is represented in CD will be called the conceptual data structure(CDS).

2. Conceptual Data Structures

We claim that it is necessary for a meaning representation of a structure to contain each and every structural concept and structural relation that is either explicitly or implicitly referred to by the structure being considered. As the primitive actions are necessary in CD, so are the structural primitives required to describe various structures of data in CD. With these structural primitives, any given structure of a data can be mapped into a construction in CD. We use these structural primitives to join similar information together so that deduction rules

for planning on a data need not be written for every individual surface structure of the data, but rather the deduction rules can be written for these primitives. Thus we will have two kinds of primitives in CD, namely, action primitives and structural primitives.

Structured objects(or simply structures) are objects that have several components. On the other hand, the objects which has no components will be called elements. The components themselves can, in turn, be structures or elements.

A structure is defined principally by describing the way in which the object and its components are interrelated. To represent this interrelationship explicitly, we adopt structors and links.

The structural primitives are:.

1. Structor: That part of a structure which gives both its basic shape and support to it.
2. Link: That part of a structure which describes the interrelationship between its structor and its component.
3. Element: The object that has no components.
4. Structure: The object that may have several components.

3. An example: The conceptual data structure of a list

In Prolog, a non-empty list, [H|T], is a structure that has two components: the head H and tail T. In general, the head can be anything but the tail has to be a list. The empty list is written as a Prolog atom, []. The structure of [1|[]] is represented in CDwith the structural primitives as shown in Fig.1.

4. Deriving Data Manipulation Plans from Structure of Data

We now study how the operation plan on a data, which achieves a given goal operation on it, can be derived from its conceptual structure. Consider the problem of planning for the following goal:

Planner wants to get itself in the rear of [1|[]].

This can be represented in CD as follows:

D-Prox(Planner, Planner, rear of [1|[]]).

where D-Prox(X,Y,Z) means that X wants Y to be at location Z. The most obvious technique to attain this goal is for Planner to move itself directly to the rear of [1|[]]. In Fig.2 this is represented in CD.

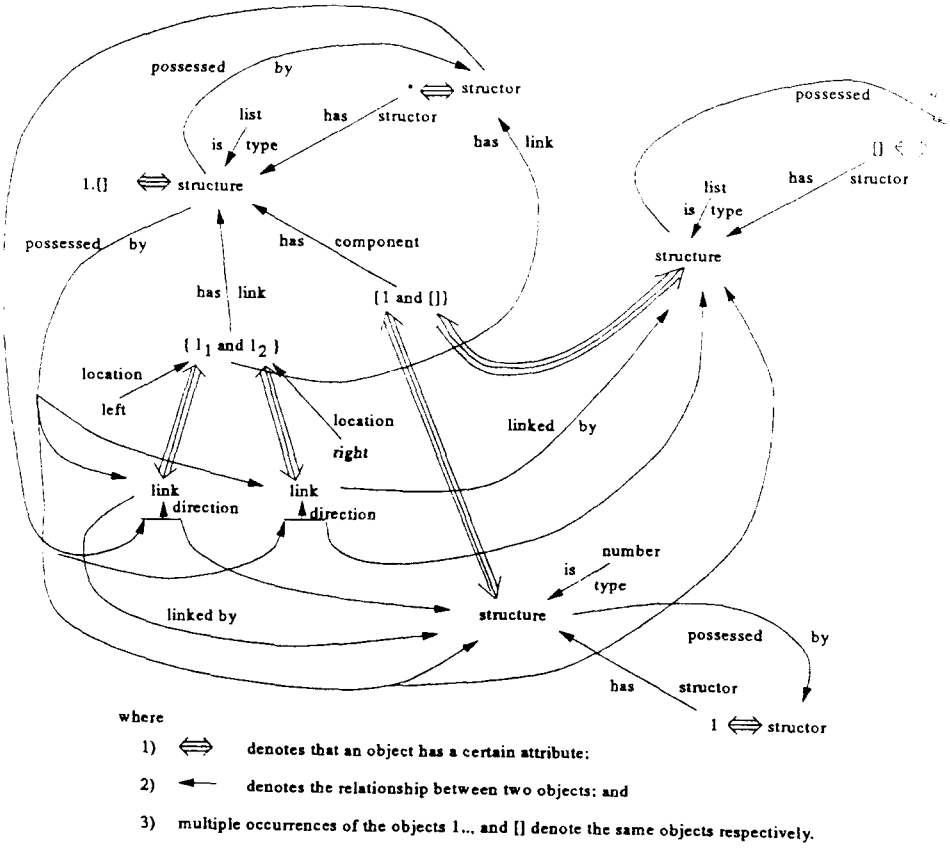
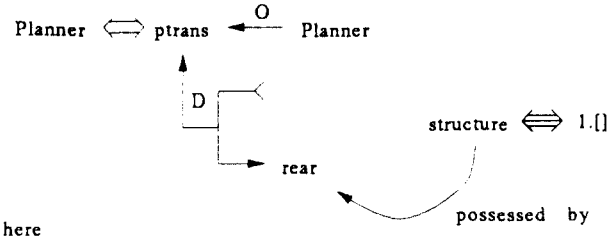


Fig.1. The conceptual data structure of a list 1.[ ].

This goal can be specified further based on the following inference rule:.

If an actor wants to do a *ptrans* to a location in a structure then that actor has to be at the structure first.



where

1) ptrans is the primitive action that denotes the transfer of the physical location of an object;
2) ⟺ denotes that an actor acts;
3) O denotes the object of an action;
4) D denotes the direction of an object within an action; and
5) possessed by denotes that an object is possessed by another object.

Fig.2. Planner moves itself to the rear of 1.[].

Thus the more specified goal is shown in Fig.3.

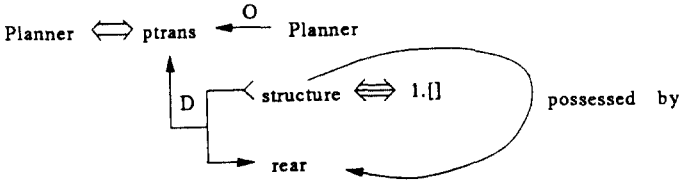


Fig.3.Planner moves itself from the structure 1.[] to the rear of it.

Note that goals may progress into one another [6]. So the goal above can progress based on the following inference rule:

If an actor wants to do a *ptrans*
from a structure to a location in that structure
then that actor has to be at the front of that structure first.

In general, when a goal progresses we will have a new goal and a plan which makes the progress. In this case, the plan which makes the progress is simply an action which is shown in Fig.4.

If we assume the front of a structure to be its structor, this action can be represented as shown in Fig.5. And the new goal will be the one in Fig.6.

Now this goal can progress again based on the following inference rule:.

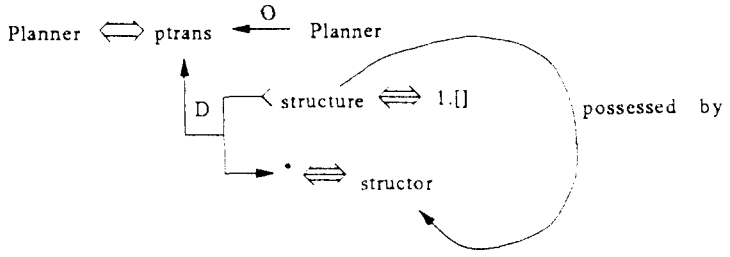


Fig.4. Planner moves itseif from the structure 1.[] to the structor of it.

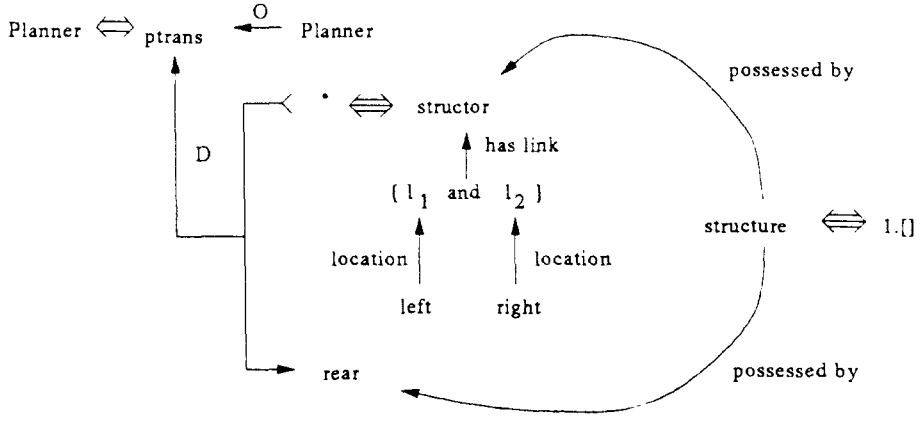


Fig.5. Planner moves itseif from the structor of the structure 1.[] to the rear of it.

If an actor wants to do a *ptrans*
from the structor of a structure to the rear of that structure
then that actor has to do that *ptrans*
via the right most link of that structor.

Different from the first case, the plan which makes this progress consists of two consecutive actions that are shown in Fig.7. And in this time the new goal will be the one in Fig.8.

We can assume the rear of a structure to be its right most substructure the structor of which is the same as its structure. With this assumption, we notice that the planner achieves the original goal.

5. Conclusion

The ultimate goal of our research is to develop a theory which explains the fact that the behavior of a system can be derived from its structure. In the meantime, we have been studying both how and why the operation sequence on a data can be

derived from its structure, which achieves a goal operation on it. In particular, we have described in this paper a minimal system of primitive concepts for expressing

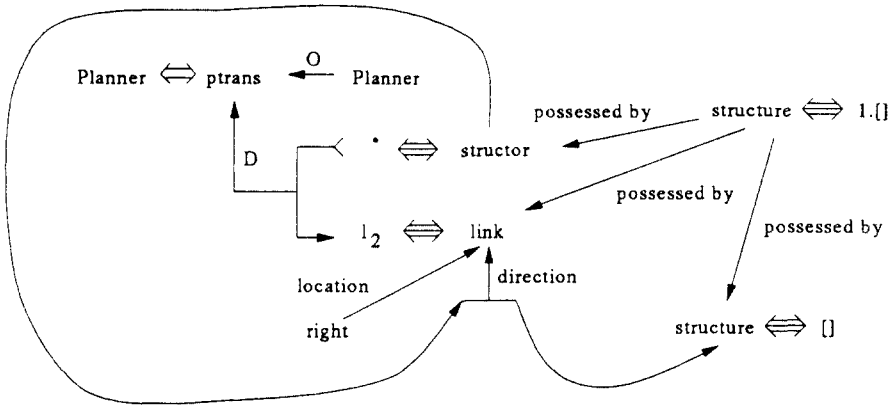


Fig.6(a). Planner moves itself directly from the structor of the structure 1.[] to the rightmost link of that structor.

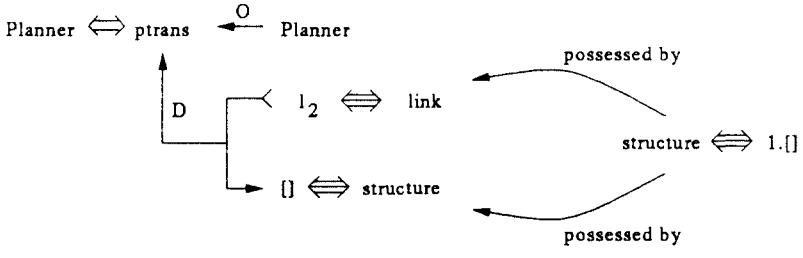


Fig.6. Planner moves itself from the structor of the structure 1.[]

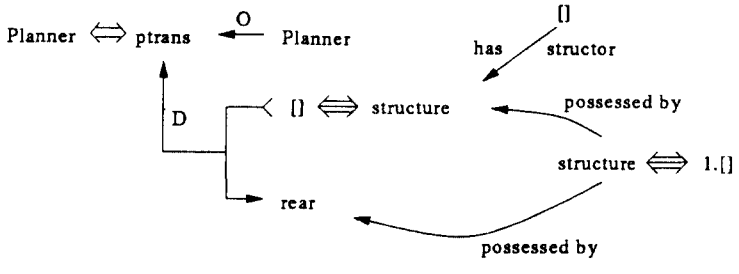


Fig.7. Planner moves itself from the substructuer[] of the structure 1.[]

conceptualizations underlying the structure of a data and have outlined a method for planning on a data which utilizes such conceptualizations. Although a complete planning system based on the approach presentedhas not been developed, we believe that this is a new conceptual approach to automatic programming.

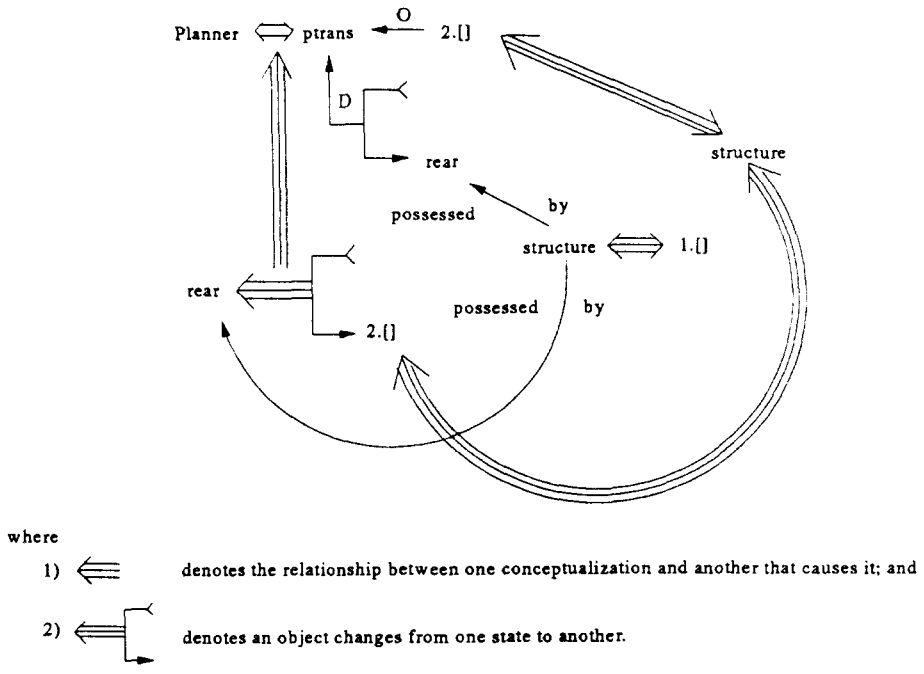


Fig.8. Planner appends 2.[] to 1.[].

References

[1] Brodie, M.L. and Ridjanovic, D., On the design and specification of database transactions, in: M.L. Brodie, J. Mylopoulos, and J.W. Schmidt (Eds.), On conceptual modelling(Springer-Verlag, New York, 1984).

[2] Jackson, M.A., Principles of program design, Academic Press, New York, 1975.

[3] Kuipers, B., Commonsense reasoning about causality: Deriving behavior from structure, in: D.G. Bobrow (Ed.), Qualitative reasoning about physical systems(MIT Press, Cambridge, MA, 1985).

[4] Orr, K.T., Structured systems development, Yourdon Press, New York, 1977.

[5] Schank, R.C., Conceptual dependency: A theory of natural language understanding, Cognitive Psychology, Vol. 3, No. 4, pp. 552-631, October 1972.

[6] Schank, R.C. and Abelson, R.P., Scripts, Plans, Goals, and Understanding, Lawrence Erlbaum Associates, Hillsdale, N.J., 1977.

[7] Warnier, J.-D., Logical construction of programs, Van Nostrand Reinhold Company, New York, 1974.

[8] Wilensky, R., Planning and Understanding: A computational approach to human reasoning, Addison-Wesley Publishing Company, Inc., 1983.