PAPER Special Issue on Intelligent CAI and Hypermedia

Load-Oriented Tutoring to Enhance Student's Explanation Understanding ——An Explanation Planner and a Self-explanation Environment——

Akihiro KASHIHARA†, *Member*, Koichi MATSUMURA†, *Nonmember*, Tsukasa HIRASHIMA† *and* Jun'ichi TOYODA†, *Members*

SUMMARY This paper discusses the design of an ITS to realize a load-oriented tutoring to enhance the student's explanation understanding. In the explanation understanding, it is to be hoped that a student not only memorizes the new information from an explanation, but also relates the acquired information with his/her own knowledge to recognize what it means. This relating process can be viewed as the one in which the student structures his/her knowledge with the explanation. In our ITS, we regard the knowledge-structuring activities as the explanation understanding. In this paper, we propose an explanation, called a load-oriented explanation, with the intention of applying a load to the student's knowledge-structuring activities purposefully. If the proper load is applied, the explanation can induce the student to think by himself/herself. Therefore he/she will have a chance of gaining the deeper understanding. The important point toward the load-oriented explanation generation is to control the load heaviness appropriately, which a student will bear in understanding the explanation. This requires to estimate how an explanation promotes the understanding activities and how much the load is applied to the activities. In order to provide ITS with the estimation, we have built an Explanation Effect Model, EEM for short. Our ITS consists of an explanation planner and a self-explanation environment. The planner generates the load-oriented explanation based on EEM. The system also makes a student explain the explanation understanding process to himself/herself. Such self-explanation is useful to let the student be conscious of the necessity of structuring his/her knowledge with the explanation. The self-explanation environment supports the student's self-explanation. Furthermore, if the student reaches an impasse in self-explaining, the planner can generate the supporting explanation for the impasse.

key words: *load-oriented tutoring, load control, explanation, self-explanation, planning, learning motivation, and impasse-driven learning*

1. Introduction

Explanation activity can be viewed as a task to make a hearer understand the explainer's explanation. The attempt to realize the task with computer is an important subject for intelligent man-machine communication systems, especially for ITS [4], [5], [10]. This paper describes the design of an ITS to enhance the student's explanation understanding.

Manuscript received July 29, 1993.
Manuscript revised September 21, 1993.

† The authors are with The Institute of Scientific and Industrial Research, Osaka University, Ibaraki-shi, 567 Japan.

It is generally desirable that a student can understand the tutor's explanation with no stress [13]. In order to afford this understanding, the tutoring systems need to explain with the intention of removing or lightening the cognitive load that the student will bear in the understanding activities [1], [5], [11], [14], [15]. From a tutoring point of view, on the other hand, it is necessary to apply a load purposefully for inducing a student to think. In this case, it is apprehended that the student loses his/her motivation for the understanding on the ground that the load is too heavy for him/her. However, if the proper load, which the student can overcome by himself/herself, is applied, he/she would pay more attention to the explanation. Therefore, he/she will have a chance of gaining a deeper understanding of it. A student, moreover, will often reach impasses owing to the load. A support that enables the student to overcome the impasses allows him/her to understand more impressively [6], [7], [12]. Consequently, in order to enhance the student's explanation understanding, it is considered necessary for the tutoring systems to generate explanations by estimating the student's cognitive load. In this research, tutoring by such explanation method is referred to as load-oriented tutoring.

First in order to realize the load-oriented tutoring, it is necessary to represent the cognitive load. This requires to assume the explanation understanding process. This research regards the explanation understanding as the process in which a student builds up a knowledge structure by relating the explanation with his/her own knowledge. On this assumption, the load can be represented as the cost of building up the knowledge structure. Second it is necessary to generate an explanation, called a load-oriented explanation, with the intention of applying a load to the student's understanding activities. The important point toward the generation is to vary the load heaviness appropriately, which a student will bear. In order to realize the load control, we need a model to estimate how an explanation promotes the understanding activities and how much the load is applied to the activities. In order to provide ITS with the estimation, we have already built an Explanation Effect Model, EEM for short [6],

[8].

This paper describes an ITS implementing the load-oriented tutoring. Our ITS consists of an explanation planner and a self-explanation environment [8]. The planner generates the load-oriented explanation based on EEM. The system also makes a student explain the explanation understanding process to himself/herself. Such self-explanation is useful to let the student be conscious of the necessity of structuring his/her knowledge with the explanation [3]. The self-explanation environment supports the student's self-explanation activities. Furthermore, if the student reaches an impasse in self-explaining, the planner can generate the supporting explanation to give a way out of the impasse.

2. Explanation Understanding

2. 1 A Consideration of Explanation Understanding

In the explanation understanding, it is to be hoped that a student not only acquires (or memorizes) the new information from a given explanation, but also recognizes what the acquired information means. This recognition requires the student to relate the acquired information with his/her own knowledge. As a result of the relating process, he/she will build up a knowledge structure. In this paper, we regard the process, in which the student structures his/her knowledge with the explanation, as the explanation understanding process.

In general, there are various directions in which a student structures his/her knowledge. However it is difficult for ITS to trace his/her knowledge-structuring processes spread out in all directions. Moreover, the explanation alone cannot always facilitate the student's knowledge-structuring activities. Consequently, in order to support the student's explanation understanding, it is necessary to restrict the directions of knowledge-structuring and make a student be conscious of the necessity of knowledge-structuring. These restrictions on the explanation understanding are considered valid in respect of tutoring.

We currently focus on a certain process in which the knowledge structure to be represented as IS-A hierarchical network is constructed. Moreover, our ITS beforehand sets a network as an understanding goal, and then makes him/her explain the process that he/she composes the network from the given explanation. Such self-explanation allows our ITS to recognize the student's knowledge-structuring activities, and besides is useful to facilitate his/her activities.

2. 2 Explanation Understanding Representation in Our ITS

Based on the above consideration, this section describes the framework of explanation understanding in our ITS.

2. 2. 1 Knowledge Structure

This research deals with the knowledge structure, which a student will build up from explanations, to be represented as IS-A hierarchical network. The node in the network indicates an object and is represented by a simplified frame [9] in which its name and attributes with values are described. The attribute inheritance is not considered, since the network is used to facilitate the student's knowledge-structuring activities as discussed in Sect. 6. The link shows the relationship between nodes. There are three pre-defined relationships as follows: 'is a subclass of' and its converse 'has a subclass,' and 'difference'. These links respectively possess the generalized attributes, the specialized attributes and the different attributes between nodes. A network in Fig. 1(a), for example, shows a knowledge structure about computer vocabularies, in which the nodes represent knowledge about vocabularies and the links show the relationships between the vocabularies.

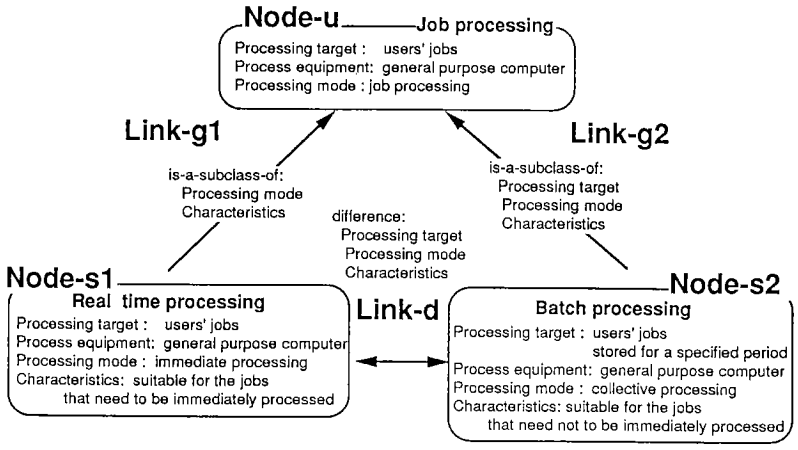
2. 2. 2 Explanation

In this research, an explanation is expressed in text form, and is represented as explanation unit sequence. Each unit in the explanation gives a description of what attributes and values a network component (node or link) has. Therefore the unit can be formally identified with the component. For example, the text (in **line(1)**) in Fig. 1(b) is composed of two explanation units: **Node-s1** and **Link-g1**. Currently we consider the unit a semantic primitive of the text. Each explanation unit comprises some sentences.

2. 2. 3 Explanation Understanding Process

Based on the above representation, the knowledge-structuring corresponds to the network composing activities such as forming nodes and linking the nodes from the explanation unit sequence. As shown in Fig. 2, moreover, the activities are divided into three phases: (1) explanation unit extraction, (2) related knowledge retrieval, and (3) network composing.

In the explanation unit extraction phase, the given text is divided into some explanation units. This division requires to pay attention to the semantic relationships between sentences in the text. In the related knowledge retrieval phase, the knowledge, which is related to the extracted explanation units, is derived from the already acquired knowledge. We assume that the acquired knowledge is represented in the same manner as the network and that the knowledge retrieval is executed by a node or link. The



(a) Goal Network

<SYSTEM> The real time processing method aims at processing user's jobs. This method uses the general purpose computer. The jobs are immediately processed after they are inputted in the computer. This method is also suitable for the jobs that need to be immediately processed. (**Node-s1**) The real time processing method is a kind of the job processing method. In the job processing method, it is not considered the immediate processing and whether the jobs need to be immediately processed. (**Link-g1**)
Explanation unit sequence....(Node-s1 Link-g1). (1)

<SYSTEM> Explain the vocabularies of batch processing method, real time processing method, and job processing method. Moreover, explain the relationships among three vocabularies. (2)

<STUDENT> What is the difference between batch processing method and real time processing method? (3)

<SYSTEM> The batch processing method aims at processing the user's jobs stored for a specified period, and collectively processes the jobs. Therefore this processing method is suited for the jobs that need not to be immediately processed. On the other hand, the real time processing method aims at processing user's jobs, and immediately processes the jobs after they are inputted in the computer. Therefore this processing method is suitable for the jobs that need to be immediately processed.
Explanation unit sequence....(Link-d). (4)

(b) Sample Dialogue

Fig. 1 An example of load-oriented tutoring.

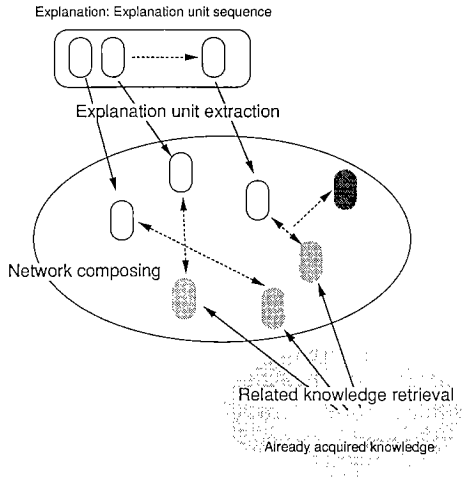


Fig. 2 Explanation understanding process.

derived knowledge may facilitate the explanation unit extraction. In the network composing phase, the extracted explanation units are related with the retrieved knowledge. As a result of the relating, the IS-A hierarchical network is built up. This network is composed of the extracted explanation units, the retrieved knowledge and the network components that are newly formed in the relating activities.

A student will bear the cognitive load in each phase. Our ITS currently deals with the load in the related knowledge retrieval and the network composing phases. We do not also consider the mistaken network composing, i.e., mistaken understanding.

3. Overview of Load-oriented Tutoring

This section gives the overview of the load-oriented tutoring in our ITS. Figure 3 shows the system

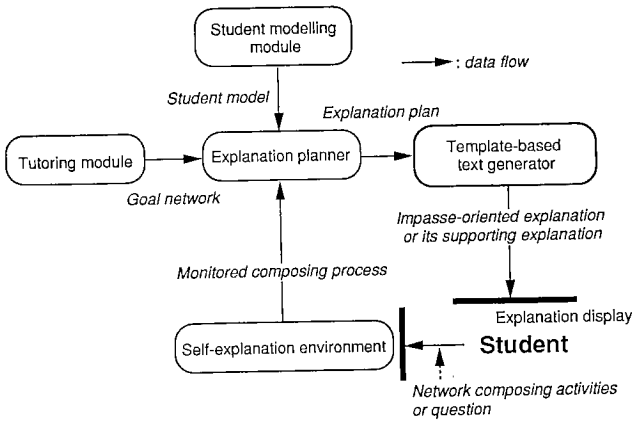


Fig. 3 System overview.

configuration. Our ITS enhances the student's explanation understanding in the following manner. First, the tutoring module sets a goal network. This network offers the knowledge structure that the system expects the student to build up finally. The tutoring goal is to let the student compose the network. The tutoring module performs the setting of such goal. (For brevity we omit a detailed discussion of this module in this paper.) Second in order to apply a load to the network composing activities, the explanation planner generates a plan for load-oriented explanation sensitively to the student model given by the student modelling module. The generated plan is translated into the text by the template-based text generator. Third the system presents this text to the student and requires him/her to compose the goal network. Since the load-oriented explanation does not give the sufficient description of the goal network, the student needs to complement the deficiencies by relating the explanation with his/her own knowledge. Lastly, if the student reaches impasses in self-explaining, the planner uses the network composing process monitored by the self-explanation environment to generate the supporting explanations for the impasses.

3. 1 Framework of Load Control

The student will bear a load when he/she tries to complement the network components that are not described by the load-oriented explanation. We assume that the heaviness of the load is in proportion to the number of the complemented components. The number, $N_{complement}$, is represented as follows: $N_{complement} = N_{total} - N_{ex\text{-}units}$, where N_{total} is the total number of the components in the goal network, and $N_{ex\text{-}units}$ is the number of the components described by the load-oriented explanation. Our ITS varies $N_{ex\text{-}units}$ to control $N_{complement}$. The system briefly controls the network composing load by varying $N_{ex\text{-}units}$. For example, the system reduces the explanation units to apply a heavier load. However $N_{complement}$ may include the number of the components the student has already acquired. Therefore the load heaviness will lighten in some degree.

In order to generate the load-oriented explanation appropriately, it is necessary to reduce $N_{ex\text{-}units}$ within the bounds of possibility that a student composes the goal network. This control requires to estimate how the network components can be complemented from an explanation unit and how much the load is applied to the complement activities. We have proposed EEM to provide our ITS with the estimation referred to as the explanation effect. Our explanation planner generates the load-oriented explanation based on EEM.

3. 2 Framework of Self-Explanation

Looking at the given load-oriented explanation, a student composes the network. In the composing process, he/she forms a node by explaining what attributes with values in the node are. He/she also forms a link by explaining what attributes in the link are. We consider that the student's knowledge-structuring is expressed as such self-explanation activities. Tracing the student's self-explanation process, our self-explanation environment tries to recognize his/her knowledge-structuring.

3. 3 Example of Load-oriented Tutoring

Here we provide a concrete example. Fig. 1(b) shows a sample dialogue with our ITS. We deal with the explanations about computer vocabularies. First, the goal network shown in Fig. 1(a) is selected. In this dialogue, we assume that a student has already acquired knowledge about batch processing (**Node-s2**), i.e., he/she can self-explain **Node-s2**. It is also assumed that the student has known all conceptions indicating the attributes and values in the goal network. Second, our ITS gives the load-oriented explanation (line(1)). This explanation has the explanation unit sequence as follows: the node of real time processing (**Node-s1**) and 'is a subclass of' link between real time processing and job processing (**Link-g1**). However it does not give the description of the following network components: **Node-s2**, **Node-u**, **Link-d** and **Link-g2**. Therefore the student needs to complement these components. The system expects the student to bear a load in complementing the components. Third, our ITS requires him/her to compose the goal network (line(2)). Since he/she has reached the impasse in complementing **Link-d**, i.e., relating **Node-s1** with **Node-s2**, he/she asks for the relationship between these nodes (line(3)). Then the system provides the supporting explanation that describes **Link-d** (line (4)).

We now discuss EEM and our ITS in more detail.

4. Explanation Effect Model: EEM

EEM estimates how the knowledge structure to be represented as IS-A hierarchical network is constructed from an explanation unit sequence. The explanation effect is embodied as the scope in which the network components can be formed by means of an explanation unit, and besides the load applied to the forming activities is described. These descriptions are provided independent of a specific domain.

4. 1 Explanation Effects

4. 1. 1 Direct Effect and Indirect Effect

There are two explanation effect types: direct effect and indirect effect. The direct effect means that a network component described by a given explanation unit is formed. The indirect effect is extended from the direct effect and means that an explanation unit helps to form some network components that are adjacent to the component formed as the direct effect.

The scope of the direct effect is called DE-Scope; the scope of the indirect effect, IE-Scope. IE-Scope comprises the network components that a student will newly form by relating the given explanation with his/her own knowledge. Expecting IE-Scope, our ITS generates the load-oriented explanation. The width of the expected IE-Scope influences the heaviness of the load that a student will bear. EEM currently limits the width to the scope directly connected with the DE-Scope.

Figure 4 shows the DE-Scope and IE-Scope of the load-oriented explanation (line(1)) in Fig. 1(b). The system expects that the explanation unit describing **Node-s1** helps the student to form **Node-s1** (in **DE-Scope1b**) directly and to form **Link-d** (in **IE-Scope1b**) indirectly. Similarly, the explanation effects of the unit describing **Link-g1** are indicated by **DE-Scope2a** and **IE-Scope2a**. We below explain how to estimate the IE-Scope of the explanation unit.

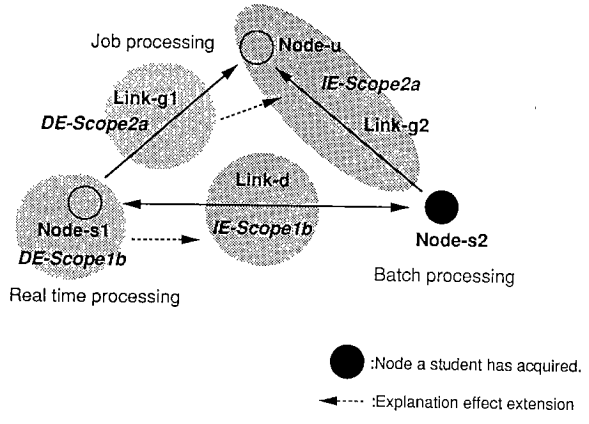


Fig. 4 DE-scope and IE-scope.

4. 1. 2 Explanation Effect Schema

Even though the same explanation unit is given, the indirect effect changes in case the context is different. Accordingly, we have schematized the explanation effect as follows: "**Context+Explanation_unit → IE-Scope.**" **Context** is provided by the student model representing the nodes and links that the student has already acquired. **Explanation_unit** prescribes DE-Scope.

Based on the explanation effect schema, we have currently described concrete several effects. The IE-Scope of each explanation effect is limited to the adjacency of the DE-Scope and the acquired knowledge. We also confine the relationship between **Context** and **Explanation_unit** in which a node is indirectly formed. As shown in Fig. 5, the node is formed only when the adjacent (the closest upper, closest lower or brother) node is already acquired and the link between two nodes is explained. Table 1 shows the explanation effect descriptions. For example, the first line from the top of Table 1 (indicated by *) describes the following effect. First let us assume that some nodes in the same hierarchical level have been already acquired. Then from the explanation unit of a 'is a subclass of' link between one of their nodes and their closest upper node, it can be expected that the upper node is formed. Moreover it can be expected that all the other 'is a subclass of' links are formed. These explanation effects are confirmed by the observation of how average students structure their knowledge in explanatory dialogues.

4. 1. 3 Explanation Effect Types

The explanation effects in Table 1 are divided into three types according to the direction from DE-Scope to IE-Scope. Figures 6(a), (b) and (c) show that the explanation effects extend respectively in bottom-up manner, in top-down manner, and in level-sideways manner. These effects are respectively called bottom-up effect, top-down effect and level-sideways effect. For example, in Fig. 4, the extension from **DE-Scope1b** to **IE-Scope1b** shows level-sideways effect. Similarly,

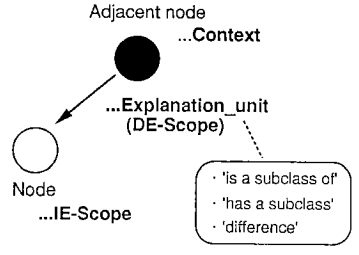
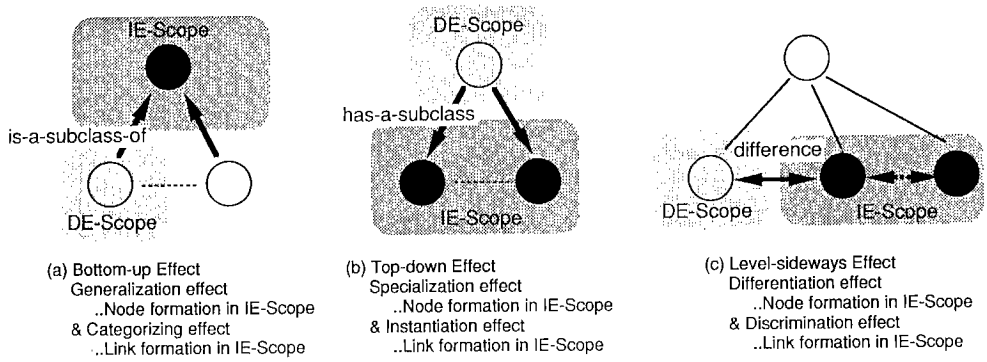


Fig. 5 Node formation in IE-scope.

Table 1 Explanation effect descriptions.

	Context	Explanation_unit	Components in IE-Scope	Indirect_effect_type
*	some nodes in the same level	a 'is a subclass of'	the upper node & the other 'is a subclass of'	Generalization&Categorizing
	a node	the lower node	'is a subclass of'	Categorizing
	a node & the lower node	'has a subclass'	'is a subclass of'	Categorizing
	a node	'has a subclass'	the lower node	Specialization
	a node	the upper node	'has a subclass'	Instantiation
	a node & the upper node	'is a subclass of'	'has a subclass'	Instantiation
	a node	'difference'	the other node in the same level	Differentiation
	some nodes in the same level	their upper node	'difference'	Discrimination
	a node	the other node	'difference'	Discrimination



(a) Bottom-up Effect
Generalization effect
..Node formation in IE-Scope
& Categorizing effect
..Link formation in IE-Scope

(b) Top-down Effect
Specialization effect
..Node formation in IE-Scope
& Instantiation effect
..Link formation in IE-Scope

(c) Level-sideways Effect
Differentiation effect
..Node formation in IE-Scope
& Discrimination effect
..Link formation in IE-Scope

Fig. 6 Explanation effect types.

the extension from **DE-Scope2a** to **IE-Scope2a** shows bottom-up effect.

The bottom-up effect, moreover, is divided into two types: generalization effect as indirect node formation and categorizing effect as indirect link formation. The generalization effect means that the closest upper node of the DE-Scope is formed. The categorizing effect means that a 'is a subclass of' link between the DE-Scope and the upper node is formed. The first line from the top of Table 1 is an example of the generalization effect and categorizing one. Similarly, the top-down effect is divided into specialization one and instantiation one; the level-sideways effect, differentiation one and discrimination one.

4. 2 Load Heaviness

Based on the above explanation effect schema, the network to be formed is composed of a component in the DE-Scope, components in the IE-Scope and already acquired components. EEM currently assumes that the component in the DE-Scope can be formed with no load. EEM represents the load heaviness as the cost of complementing the remaining components. The cost is in proportion to the number of the components. The cost of complementing one component is also in proportion to the number of the attributes in the component. From this point of view, EEM currently describes the load as the number of attributes in the complemented components. EEM also assumes that each network component requires its own complement cost. We discuss each cost below:

- **Cost of complementing a component in IE-Scope**

EEM assumes that the complement cost changes according to whether the component is a node or link, and the direction to which the component is complemented from DE-Scope. First, in order to form a node, it is necessary to generalize (specialize or discriminate) attributes (or values), although link formation requires to compare the attributes in a node with the ones in the other node. Since it is considered that the generalization (specialization or discrimination) is harder than the comparison, the cost for node formation is more than that for link formation. Second, the bottom-up complement is considered harder than the top-down complement. The top-down complement is also considered harder than the level-sideways complement.

- **Cost of complementing a component that a student has acquired**

EEM assumes that the complement cost changes according to the capability that a student brings back

his/her memory. In other words, the complement by the poor capability is harder than that by the high capability.

The above assumptions have not been verified yet, but the load-oriented explanation based on EEM is considered useful from an intelligent tutoring point of view. We currently regard EEM as a model to provide the necessary estimation for planning the load-oriented explanation.

5. Explanation Planner

This section discusses the methods to generate the load-oriented explanation based on EEM and the supporting explanation. These methods contribute to the generic explanation-planning mechanism to enhance the student's explanation understanding in which the IS-A hierarchical network is built up. In other words, our ITS is applicable to any domain, of which the knowledge structure built up from explanations can be represented as IS-A hierarchical network.

5.1 Planning Architecture

Before planning, our ITS makes a student compose the goal network with only all node names included in the network. (The sample dialogue in Fig. 1 omits the student modelling.) From the composition result, the student modelling module uses overlay modelling technique [2], [16] to identify which of the components the student has acquired (self-explained), and generates an initial student model. The student model is represented as the network overlaid with the goal network.

Figure 7 shows the explanation planning process in our ITS. First, the load-oriented explanation planning is executed. The planner generates a plan in order to change the initial student model into the goal network (**Load-oriented explanation plan generation**). Then expecting the explanation effect by each explanation unit, the planner makes the explanation unit sequence. Next, the planner estimates the heaviness of the load that the generated plan will apply to the student's knowledge-structuring activities (**Load estimation**). The estimated heaviness is compared to a threshold (**Load control**). If it exceeds the threshold, the planner modifies the plan (**Replanning**); if not, the plan is translated into natural language by the template-based text generator shown in Fig. 3. The generated load-oriented explanation is presented to a student.

Second, if an impasse occurs when the student composes the goal network after the load-oriented explanation, the supporting explanation planning is executed. Then the planner uses the information monitored by the self-explanation environment to generate a plan for the supporting explanation (**Supporting explanation plan generation**).

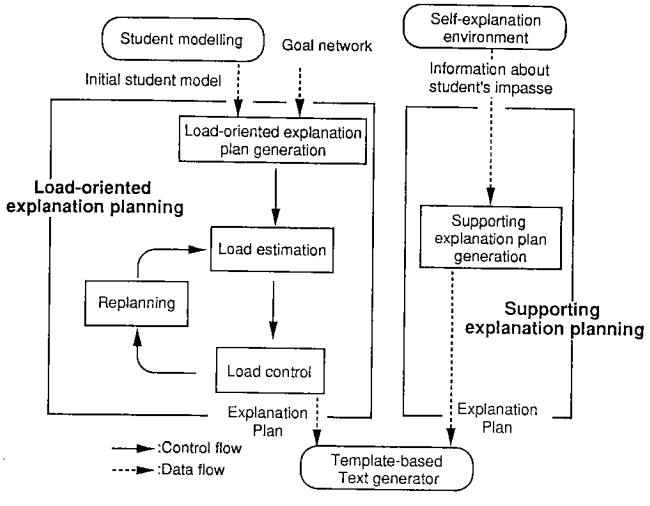


Fig. 7 Explanation planning process.

5.2 Load-oriented Explanation Planning

5.2.1 Planning Operators

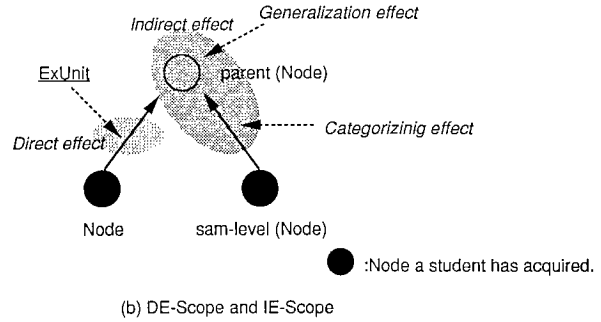
In our planner, the planning operators are divided into two types: basic operators and explanation strategies. The basic operators derive explanation units from the goal network, and provide what explanation effects are expected from the units. The explanation strategies decide the order of searching for basic operators.

• **Basic operators**

Each basic operator contains a **Goal**, which represents an explanation effect type provided by the operator; a **Constraint** list, which is a context that the operator is applicable; **ExUnit**, which is an explanation unit extracted by the operator; **Direct_effect** and **Indirect_effect**, which denote the effects of the **ExUnit**. Figure

```
(Basic-op1
   (Goal          : bottom-up)
   (Constraint    : (node(Node) same-level(Node)))
   (ExUnit        : ex-unit(is-a-subclass-of (Node parent(Node))))
   (Direct_effect : formation-link(Node parent(Node)))
   (Indirect_effect : (formation-node(parent(Node))
                      formation-link(same-level(Node) parent(Node))))
)
```

(a) Operator Description



(b) DE-Scope and IE-Scope

Fig. 8 A basic operator for providing bottom-up effect.

8(a) shows a basic operator that provides the bottom-up effect in the first line from the top of Table 1. This operator derives the explanation unit describing the 'is a subclass of' link between **Node** and its upper node (**parent**(**Node**)). The operator also provides the explanation effects shown in Fig. 8(b).

• **Explanation strategies**

The explanation strategies also search basic operators to extend the explanation effect from the node that a student has acquired (called starting node). If there is no starting node in the initial student model, it is necessary to make up a starting node. Then the system explains a node to the student with no load, and makes him/her self-explain the node. Conversely if there are some starting nodes, the lower node is selected.

In order to generate the appropriate load-oriented explanation, it is efficient to first generate a plan that applies as heavy load as possible and subsequently to decrease the load to a threshold. In order to implement this planning manner, the explanation strategies follow the two search preferences: node formation preference and direction preference. The node formation preference provides that the basic operators for providing indirect node formation have preference to the ones for indirect link formation. This is supported by the assumption that the node formation load is heavier than the link formation load as discussed in Sect. 4. 2. Similarly the direction preference provides that the directions of the explanation effect extension are selected in priority order here: bottom-up, top-down, and level-sideways.

5. 2. 2 Plan Generation

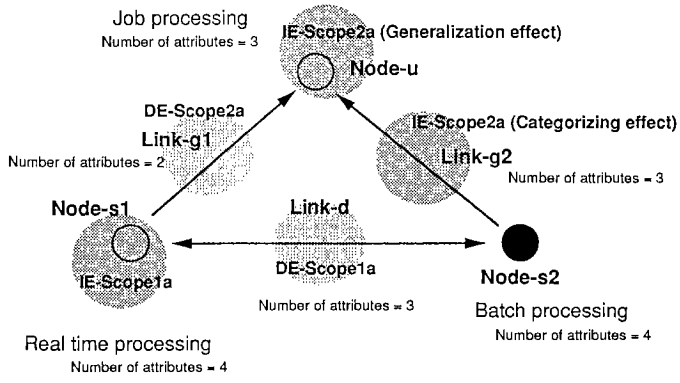
This section describes the planning process with an example shown in Fig. 9. This example represents the planning for the sample dialogue in Fig. 1(b). In this planning, an initial student model provides that only **Node-s2** has been acquired. First, following node formation preference, the planner decides the direction of the explanation effect extension. If it is not possible to extend the effect in any directions, the planner considers the link formation. In this example, the bottom-up extension from the starting node, i.e., **Node-s2**, is not possible, since the context does not satisfy the constraints of the basic operators for bottom-up effect. (The top-down extension is not considered.) Then the planner chooses the level-sideways extension from **Node-s2** to **Node-s1**, and searches a basic operator for differentiation effect. As a result, the explanation unit (**ExUnit-1a**) is replaced into **Plan-A** shown in Fig. 9(a). **ExUnit-1a** describes **Link-d**. The **DE-Scope** and **IE-Scope** are indicated by respectively **DE-Scope1a** and **IE-Scope1a** as shown in Fig. 9(b).

Second, the planner assumes that the above differentiation effect was obtained, i.e., the student

```
Plan-A:
(ImpassePlan
   (ex-unit(difference (RealTimeProcessing  BatchProcessing)) ..... ExUnit-1a
      formation-link(RealTimeProcessing BatchProcessing)
      formation-node(RealTimeProcessing))
   (ex-unit(is-a-subclass-of (RealTimeProcessing  JobProcessing)) ..... ExUnit-2a
      formation-link(RealTimeProcessing JobProcessing)
       formation-node(JobProcessing)
      formation-link(BatchProcessing JobProcessing))
)

Plan-B:
(ImpassePlan
   (ex-unit(node (RealTimeProcessing))                                  .... ExUnit-1b
      formation-node(RealTimeProcessing)
      formation-link(RealTimeProcessing BatchProcessing))
   (ex-unit(is-a-subclass-of (RealTimeProcessing JobProcessing))     ..... ExUnit-2a
      formation-link(RealTimeProcessing JobProcessing)
       formation-node(JobProcessing)
      formation-link(BatchProcessing JobProcessing)) )
```

(a) Explanation Plans



(b) Explanation Effect Extension

Fig. 9 An example of explanation planning.

could form **Link-d** and **Node-s1**, and then recursively searches the next basic operator. Then the operator shown in Fig. 8 is chosen for the bottom-up extension from **Node-s1** to **Node-u**. As a result, **ExUnit-2a** is replaced into **Plan-A**. **ExUnit-2a** describes **Link-g1**. As shown in Fig. 9 (b), the DE-Scope is indicated by **DE-Scope2a**. The IE-Scope is also indicated by **IE-Scope2a** including **Node-u** as generalization effect and **Link-g2** as categorizing effect.

5. 2. 3 Load Estimation

This section embodies the load heaviness based on the discussion in Sect. 4. 2. First let us consider the load applied by an explanation unit. $H(unit)$ represents the load heaviness. The variable N represents the total number of attributes in the network components to be formed by the unit. The variables Nd and Ni respectively represent the number of attributes in the DE-Scope and the one in the IE-Scope. The variable Nc represents the number of attributes in the components that the student has already acquired. These variables, Nd, Nc, Ni and N are related through $N=Nc+Nd+Ni$. The heaviness can be represented below on the

assumptions that the node formation is heavier than the link formation and that the load depends on the capability a student brings back his/her memory:

$$H(unit)=\frac{\{(Nin+w+Nil)+(1-r)\times Nc\}}{N}$$

where $0\leqq w\leqq 1$, $0\leqq r\leqq 1$, $Ni=Nin+Nil$.

The variables *Nin* and *Nil* respectively represent the number of attributes in the nodes and the one in the links. The variable w represents the proportion of the link formation load to the node formation load. The variable r represents the student's capability. If the student can completely bring back his/her memory with no load, then the variable r is as follows: $r=1$. On the contrary, if he/she cannot bring back at all, then r is as follows: $r=0$.

The numerator of $H(unit)$ represents the number of the attributes the student needs to complement from the *unit*. The variables w and r must be set according to the student's knowledge-structuring capability. Since it is very difficult to set them dynamically, our planner currently deals with them fixedly. However, towards the load-oriented tutoring, it is important to estimate the load heaviness by considering these variables.

Next, let us represent the load heaviness by the explanation plan. The plan includes several explanation units. If even one unit applies a heavy load, a student may lose his/her learning motivation. Accordingly the heaviness should be represented as $H(plan)=\max\{H(unit_i)\}$, where $\{H(unit_i)\}$ is a set of which each element is $H(unit_i)$ by a $unit_i$ in the plan.

5. 2. 4 Load Control

If a load is trivial, the student will easily compose the goal network. Conversely if an overload applies, he/she may lose his/her understanding motivation. Accordingly the appropriate load control is indispensable for the load-oriented tutoring.

The planner sets a threshold, referred to as *Hth*, to control $H(plan)$. The heaviness of the generated plan is compared to *Hth*. If $H(plan)\leqq Hth$, the plan is translated into natural language. If not, the plan is modified. Our system regards *Hth* as a factor that represents the student's capability of knowledge-structuring. *Hth* is updated according to his/her self-explanation discussed in Sect. 6.

5. 2. 5 Replanning

In order to decrease $H(plan)$, the planner replaces the basic operator that provides maximum heaviness among $\{H(unit_i)\}$ with a new operator. The replacement follows the node formation preference. If the replaced operator provides the indirect node formation, the planner replaces it with the operator to provide the indirect link formation. If the replaced operator provides the indirect link formation, the planner replaces it with the operator to provide the link formation directly.

5. 2. 6 Example of Load Estimation, Load Control and Replanning

We return to the example of explanation planning shown in Fig. 9. In this example, we assume the variables, $w=0.5$ and $r=0.8$. The load heaviness by each explanation unit in the first generated plan (**Plan-A**) is as follows:

$$H(ExUnit-1a)=\frac{\{4+0.5\times 0)+(1-0.8)\times 4\}}{11}$$

$$=0.44$$

where $Nin=N_{Node-s1}=4$, $Nil=0$,

$Nc=N_{Node-s2}=4$, and

$N=N_{Node-s2}+N_{Link-d}+N_{Node-s1}$

$=4+3+4=11$

$$H(ExUnit-2a)=\frac{\{(3+0.5\times 3)+0.2\times 8\}}{16}=0.38$$

where $Nin=N_{Node-u}=3$, $Nil=N_{Link-g2}=3$,

$Nc=N_{Node-s1}+N_{Node-s2}=8$, and

$N=N_{Node-s1}+N_{Node-s2}+N_{Link-g1}$

$+N_{Node-u}+N_{Link-g2}$

$=4+4+2+3+3=16$.

($N_{component}$ represents the number of the attributes in the component.)

Therefore

$$H(Plan-A)=\max\{0.44,\ 0.38\}=0.44.$$

The number 0.44 represents that the student needs to complement about 40 percent of the goal network.

If the threshold $Hth=0.4$, the planner needs to modify **Plan-A**. In this case, the operator providing **ExUnit-1a** is replaced. Since this operator provides the formation of **Node-s1**, the planner replaces it with a new operator to provide the indirect link formation. As a result, **ExUnit-1a** is replaced with **ExUnit-1b** in **Plan-B** shown in Fig. 9(a). **ExUnit-1b** describes **Node-s1** and the DE-Scope and IE-Scope respectively include **Node-s1** and **Link-d** as shown in Fig. 4. The load heaviness by **ExUnit-1b** is as follows:

$$H(ExUnit-1b)=\frac{\{0+0.5\times 3)+0.2\times 4\}}{11}=0.21$$

where $Nin=0$, $Nil=N_{Link-d}=3$,

$Nc = N_{Node-s2} = 4$, and

$N = N_{Node-s2} + N_{Node-s1} + N_{Link-d}$

$= 4 + 4 + 3 = 11.$

Therefore $H(Plan\text{-}B) < Hth$. The load-oriented explanation (line(1)) in Fig. 1(b) is generated from the **Plan-B**.

5.3 Supporting Explanation Planning

The supporting explanation is generated according to the part in which a student reached an impasse in composing the goal network. If the part is in the DE-Scope of the load-oriented explanation, the planner points out the corresponding portion of the explanation. If the part is in the IE-Scope, the planner generates the corresponding explanation unit without the load estimation.

6. Self-Explanation Environment

Figure 10 shows our self-explanation environment, which has been implemented in OpenWindows on SUN SPARCstation2.

Looking at the given load-oriented explanation, a student composes the goal network. Then it is difficult to be aware of the attribute inheritance beforehand. In order to help a student find the similarity and further difference between nodes, the network is described without the attribute inheritance.

The network composing operations are proceeded with in the "**Self Explanation**" window. A student can explain nodes and links by mouse-selecting them from a menu of network components (**Menu of components**). The attributes and their values in the nodes and links are prepared as a menu (**Attribute & Value**). This menu also includes some misleading attributes and values. The student can put down the required attribute and its value by selecting and double-clicking them. Furthermore, he/she can receive a supporting explanation to overcome the impasse by selecting a menu of questions and besides pointing to the network component that he/she would like to ask (**Menu of Question**). These functions of the environment reduce the hardness of natural language interpretation.

The self-explanation environment acquires the following information by tracing the mouse operations by the student: a sequence of network composing operations and intervals between them. Such tracing can be easily implemented by OpenWindows. The environment also recognizes the occurrence of impasses when the student has asked or an interval between the operations has been more than a given period. In case of the latter occurring, the environment asks him/her where he/she tries to compose, in order to identify the network components in which the impasses occur. If the student does not need the supporting explanation, he/she can continue to compose the network without answering this question.

Some students may also make mistakes in the network composing activities owing to their misconceptions. In order to realize more advanced load-oriented tutoring, it is necessary to cope with the misconceptions. However we currently focus on the impasses owing to the knowledge-structuring capability. Therefore our ITS does not distinguish the support for the misconception from the one for the impasses.

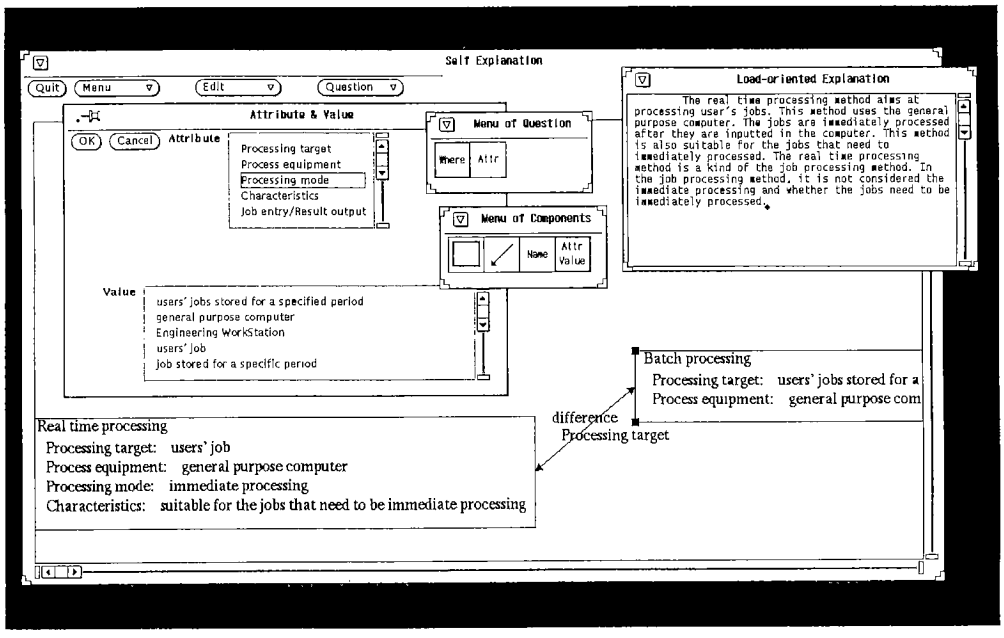


Fig. 10 An example of network composing operations.

If there are the network components that are not complemented after the student's self-explanation, the system points out them to require him/her to self-explain. If the student has built up the goal network with no impasse, our system tries to increase the threshold *Hth*, which is used for the load control for the next load-oriented explanation.

7. Conclusion

As a first step to the load-oriented tutoring to enhance the student's explanation understanding, we have designed an ITS: an explanation planner and a self-explanation environment. In this paper, the explanation understanding is regarded as the knowledge-structuring. The planner generates the load-oriented explanation by estimating the load that a student will bear in structuring his/her knowledge with the explanation. We have also proposed EEM to realize the appropriate load control. EEM provides the planner with a foundation of the estimation. The self-explanation environment helps a student to structure his/her knowledge with explanations. When the student has reached impasses in self-explaining his/her knowledge-structuring process, the planner also generates the supporting explanations. These modules make it possible to enhance the student's explanation understanding.

The current system is limited in various ways that could be addressed with future research. For example, EEM is not always a cognitive model adequate to represent the human knowledge-structuring process, and the support for the impasses owing to the student's misconception has not been considered yet. Finally, we will need to verify the assumptions in EEM, and then evaluate the effectiveness of the load-oriented tutoring, especially the method of the load estimation and load control, by testing our ITS.

Acknowledgment

The authors wish to thank the ITS research group members of I.S.I.R., Osaka University, for their valuable comments. This research is supported in part by Grant-in-Aid for Scientific Research No. 05780163 from the Ministry of Education, Science and Culture of Japan.

References

[1] Cawsey, A., "Generating Interactive Explanations," *Proc. of AAAI*, pp. 86-91, Aug. 1991.

[2] Carr, B. and Goldstein, I.P., "Overlays: A Theory of Modeling for Computer Aided Instruction," *MIT AI Lab Memo 406*, 1977.

[3] Chi, M. T. H. and VanLehn, K., "Self-explanations: How students study use examples in learning to solve problems," *Cognitive Science*, vol. 13, pp. 145-182, 1989.

[4] Kashihara, A., Hirashima, T., Nakamura, Y. and Toyoda, J., "Providing Advanced Explanation Capability in ITS for Object Understanding Support: Explanation Structure Model: EXSEL," *Trans. IEICE*, vol. J74-D-II, no. 11, pp. 1583-1595, Nov. 1991.

[5] Kashihara, A., Nishikawa, T., Hirashima, T. and Toyoda, J., "Advanced Learning Environment Based on Intelligent Explanation Capability for Object Understanding Support," *Trans. IEICE*, vol. J75-A, no. 2, pp. 286-295, Feb. 1992.

[6] Kashihara, A., Matsumura, K., Hirashima, T. and Toyoda, J., "Impasse Control for Enhancement in ITS: Impasse-oriented Explanation Planning and Self-explanation Interface," *Technical Report of JSAI*, SIG-IES-9204-5, pp. 25-30, Mar., 1993.

[7] Kashihara, A., Hirashima, T. and Toyoda, J., "Impasse-oriented Explanation Planning Based on Explanation Effect Model," *Abridged Proc. of HCI International '93*, pp. 150, Aug. 1993.

[8] Kashihara, A., Matsumura, K., Hirashima, T. and Toyoda, J., "Towards Load-oriented Tutoring for Student's Explanation Understanding," *Proc. of ICCE '93*, Dec. 1993.

[9] Minsky, M., *A framework for representing knowledge: In The Psychology of Computer Vision*," ed P. Winston, New York, McGraw-Hill, 1975.

[10] Moore, J. D. and Swartout, W. R., "Reactive Approach to Explanation," *Proc. of IJCAI*, pp. 1504-1510, Aug. 1989.

[11] Moore, J. D. and Swartout, W. R., "Pointing: A Way Toward Explanation Dialogue," *Proc. of AAAI*, pp. 457-464, 1990.

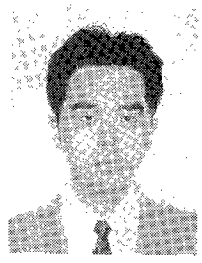
[12] So'ota, A., Kashihara, A., Hirashima, T. and Toyoda, J., "Learning Environment for Problem Solving with Diagram," *Techical Report of IPSJ SIG Notes*, vol. 93, no. 9, 93-CE-25-1, pp. 1-10, Jan. 1993.

[13] Suthers, D. D., "Answering Student Queries: Functionality and Mechanisms," *ITS '92 Lecture Notres in Computer Science number 608*, pp. 191-198, Jun. 1992.

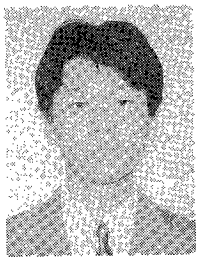
[14] Suthers, D., Woolf, B. and Cornell, M., "Steps from Explanation Planning to Model Construction Dialogue," *Proc. of AAAI*, pp. 24-30, Aug. 1992.

[15] Valley, K., "Explanation in Expert System Shells: A Tool for Exploration and Learning," *ITS '92 Lecture Notes in Computer Science number 608*, pp. 601-614, Jun. 1992.

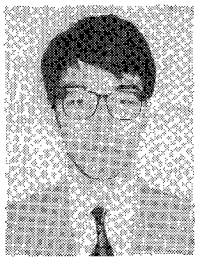
[16] Wenger, E., *Artificial Intelligence and Tutoring systems*, Morgan Kaufmann Publishers, Los Altos, 1987.



Akihiro Kashihara received the B.S. and M.S. degrees in computer science and system engineering from Tokushima University, Tokushima, Japan, in 1987 and 1989 respectively, and the Dr. of Eng. degree in information and computer sciences from Osaka University, Osaka, Japan, in 1992. He is now a Research Associate of the Institute of Scientific and Industrial Research at Osaka University. His research interests include intelligent tutoring systems, explanation planning and program understanding. He is a member of Information Processing Society of Japan, Japanese Society for Artificial Intelligence and Japan Society for CAI.



Koichi Matusmura received the B.S. degree in applied physics from Osaka University, Osaka, Japan in 1992. He is presently a master student in the Department of Applied Physics, Osaka University. His research interests include intelligent tutoring systems, explanation, and user interface.



Tsukasa Hirashima graduated in 1986 from the Dept. of Appl. Phys., Fac. Eng., Osaka University, where he obtained a Dr. of Eng. degree in 1991 and became an Assistant at the Inst. Sci. Indust. Res., Osaka University. Presently, he is engaged in research on artificial intelligence, especially in intelligent tutoring systems. He is a member of the Inf. Proc. Soc. Jap.; Soc. Artif. Intel.; and Educ. Tech. Soc.

Jun'ichi Toyoda graduated in 1961 from the Dept. of Comm. Eng., Fac. Eng., Osaka University, where he obtained a Dr. of Eng. degree in 1966 and became an Assistant in 1966 and an Assoc. Prof. in 1969 on the Fac. Eng. Sci. He became a Professor in 1982 at the Inst. Sci. Indust. Res., Osaka University. Presently, he is engaged in research on intelligent tutoring systems, natural language comprehension, image processing and document image processing. He is a member of the Inf. Proc. Soc. Jap.; Soc. Artif. Intel.; and Jap. Soc. Cogn. Sci.