数理解析研究所講究録
第 790 巻 1992 年 109-115

高階アブストラクションに基づく自然演繹証明の類推

ANALOGICAL REASONING OF NATURAL DEDUCTION PROOF
BASED ON
HIGHER ORDER ABSTRACTION

原尾　政輝（Masateru HARAO）

Department of Artificial Intelligence
Kyushu Institute of Technology, Iizuka ,820,JAPAN
TEL. 0948-29-7612　e-mail: harao@dumbo.ai.kyutech.ac.jp

Abstraction

This paper presents a framework of obtaining a natural deduction proof of a given logic formulae based on similarity among formulas under the assumption that similar formulas have similar solutions. A formulae to which a proof has already been given is called a guiding problem. From a guiding problem, a schema which is applicable to a class of similar formulas is constructed by abstraction. A schema acts as a specification of proofs and any object formulae having the same type to a schema can be obtained according to the typed proof structure. The analogical reasoning based on this idea is formalized using typed language in the framework of higher order logic. Finally, we show that this analogical reasoning procedure can be realized based on higher order unification within the computable scope.

1.Introduction

In order to realize an intelligent system on machine, one of the most important problem is to introduce a reasoning mechanism which break through the wall of present deductive theorem proving paradigm. The analogical reasoning is a mechanism to reason by finding certain similarity with some already known problem, and is considered as a most essential mechanism which supports the creative thinking of human beings. It has been proposed several kinds of models for analogical reasoning systems[6,10,13]. Among them , the reasoning system based on the generalized knowledge produced from already known formulae by abstraction is called the abstraction based analogy[2,3,4,5,8,14,16]. In this paper,an abstraction based analogical reasoning system for LK proving will be formalized as illustrated in Fig.1. Where,the proof of a guiding problem is known and this proof structure is abstracted as proof schema. Then a new formulae ? is proved using the similarity between ? and some guiding problem. By this analogical reasoning process,we can expect to reduce the nondeterministic aspects from the processing and to realize certain non-deductive reasoning.

2 LK system and natural deduction proof

2.1 LK system

The LK system is a logic system which consists of the following inference rules.

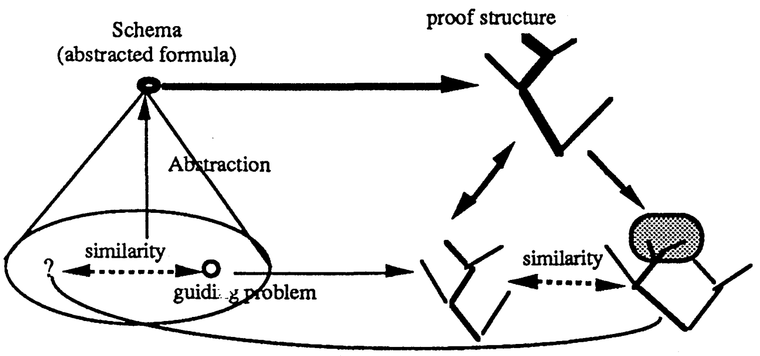

Fig.1 Proof construction by analogy

$$\text{(or-L)}\ \frac{A,\Gamma\to\Theta\quad B,\Gamma\to\Theta}{A\vee B,\Gamma\to\Theta}\qquad \text{(or-R)}\ \frac{\Gamma\to\Theta,A,B}{\Gamma\to\Theta,A\vee B}\qquad \text{(and-L)}\ \frac{A,B,\Gamma\to\Theta}{A\wedge B,\Gamma\to\Theta}\qquad \text{(and-R)}\ \frac{\Gamma\to A\quad \Gamma\to B}{\Gamma\to A\wedge B}$$

$$\text{(imp-L)}\ \frac{\Gamma\to\Theta,A\quad B,\Delta\to\Lambda}{\Gamma,A\supset B,\Delta\to\Theta,\Lambda}\qquad \text{(imp-R)}\ \frac{\Gamma,A\to\Theta,B}{\Gamma\to\Theta,A\supset B}\qquad \text{(all-L)}\ \frac{A[x:=t],\Gamma\to\Theta}{\forall xA(x),\Gamma\to\Theta}\qquad \text{(all-R)}\ \frac{\Gamma\to\Theta,A[x:=y]}{\Gamma\to\Theta,\forall xA(x)}$$

$$\text{(some-L)}\ \frac{A[x:=y],\Gamma\to\Theta}{\exists xA(x),\Gamma\to\Theta}\qquad \text{(some-R)}\ \frac{\Gamma\to\Theta,A[x:=t]}{\Gamma\to\Theta,\exists xA(x)}\qquad \text{(thin-L)}\ \frac{\Gamma\to\Theta}{A,\Gamma\to\Theta}\qquad \text{(thin-R)}\ \frac{\Gamma\to\Theta}{\Gamma\to\Theta,A}$$

$$\text{(not-L)}\ \frac{\Gamma\to\Theta,A}{\sim A,\Gamma\to\Theta}\qquad \text{(or-R)}\ \frac{A,\Gamma\to\Theta}{\Gamma\to\ \sim A,\Theta}\qquad \text{(Cut)}\ \frac{\Gamma\to\Theta,A\quad A,\Delta\to\Lambda}{\Gamma,\Delta\to\Theta,\Lambda}$$

A sequent A→A is trivially true and is called an axiom. A LK natural deduction proof is produced by applying the inference rules in nondeterministic, and can be represented by a derivation tree.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p(a)\to p(a)\quad \dfrac{q(a)\to q(a)}{q(a)\to\exists xq(x)}}{p(a),p(a)\supset q(a)\to\exists xq(x)}}{p(a),\forall x\,(p(x)\supset q(x))\to\exists xq(x)}\quad \dfrac{\dfrac{q(b)\to q(b)}{q(b)\to\exists xq(x)}}{q(b),\forall x\,(p(x)\supset q(x))\to\exists xq(x)}}{p(a)\vee q(b),\forall x\,(p(x)\supset q(x))\to\exists xq(x)}}{(p(a)\vee q(b))\wedge\forall x\,(p(x)\supset q(x))\to\exists xq(x)}}{\to p(a)\vee q(b)\wedge\forall x\,(p(x)\supset q(x))\supset\exists xq(x)}$$

Fig.2 A natural deduction proof.

A formulae is provable if there exists a proof tree whose root and leaves are labeled with the formulae and certain axioms respectively. In Fig.2, an example of LK proof of the following formulae is shown: $\to(p(a)\vee q(b))\wedge\forall x\,(p(x)\supset q(x))\supset\exists xq(x)$

2.2 Term representation of LK proof

Each inference rule is looked upon a function which maps from the assumptions given in the upper side of the rule to the conclusion given in the lower side of the rule. For example, the or-L rule corresponds to a function with the type [A, Γ→Θ]→[B,Γ→Θ]→[A∨B,Γ→Θ].

$$\frac{A,\Gamma\to\Theta\quad B,\Gamma\to\Theta}{A\vee B,\Gamma\to\Theta}\ \text{(or\_L)}$$

This can be represented as the following term:

[A∨B,Γ→Θ] = or-L([A, Γ→Θ],[B,Γ→Θ])

In the similar way, any LK proof is able to be represented by a term. In the followings, we denote the term representation of a proof for sequent Γ→Θ as *proof*(Γ→Θ), and call it as a *proof term*..

The labels of any leaves of completed proof are the axioms, that is, the sequents in the form $[A, \Gamma\to\Theta, A]$ or equivalently in the form $[A\to A]$. Thus,the proof term of each completed proof is the form such that *term*(...,*term*(A→A),...,*term*(B→B)). A proof in which some parts are not completed is called a *partial proof.* It is noted that the sequents at the leaves of partial proofs are not always axioms. For example, let us consider the proof given in Fig.3,which is a proof of Fig.2 in which the subproofs for the formulas $[p(a),\forall x(p(x)\supset q(x))\to\exists xq(x)]$ and $[q(b)\to\exists xq(x)]$ are not completed.

$$\dfrac{\dfrac{\dfrac{p(a),\forall x\,(p(x)\supset q(x))\to\exists xq(x)\qquad \dfrac{q(b)\to\exists xq(x)}{q(b),\forall x\,(p(x)\supset q(x))\to\exists xq(x)}\,(\text{thin\_L})}{p(a)\vee q(b),\ \forall x\,(p(x)\supset q(x))\to\exists xq(x)}\,(\vee\text{\_L})}{(p(a)\vee q(b))\wedge\forall x\,(p(x)\supset q(x))\to\exists xq(x)}\,(\wedge\text{\_L})}{\to(p(a)\vee q(b))\wedge\forall x\,(p(x)\supset q(x))\supset\exists xq(x)}\,(\supset\text{\_R})$$

Fig.3 A partial proof of Fig.2

A term representation of this partial proof is given as follows:

λX λY. imp-R(and-L(or-L(X,thin-L(Y))))

,where X and Y represent the partial proof for $[p(a),\forall x\,(p(x)\supset q(x))\to\exists xq(x)]$ and $[q(b)\to\exists xq(x)]$ respectively. Therefore, the partial proof given in Fig.3 implies the proof having the type

$$[p(a),\forall x(p(x)\supset q(x))\to\exists xq(x)]\to[q(b)\to\exists xq(x)]\to[\to(p(a)\vee q(b))\wedge\forall x\,(p(x)\supset q(x))\supset\exists xq(x)].$$

We can consider that the proofs whose proof tree are different only at the leaves are similar together. Basing on this idea, an analogical reasoning system will be designed.

3 Schemata for Proof Analogy

3.1 Simple Schema as proof types

We call a formulae whose proofs have already been known to be a guiding formulae or guiding problem.We assume that some guiding problems are collected as a database. A schema constructed from guiding problem g is defined as a formulae in which some predicates of g are abstracted as predicate variables. A *simple schema* is a schema which is constructed from g by simply replacing several predicates appearing in g with predicate variables. For example, let g be a formulae such that

$$g=[p(a)\vee q(b)]\wedge\forall x(p(x)\supset q(x))\supset\exists x.q(x).$$

$$\dfrac{\dfrac{\dfrac{\dfrac{P(a)\to P(a)\qquad \dfrac{Q(a)\to Q(a)}{Q(a)\to\exists xQ(x)}}{P(a),P(a)\supset Q(a)\to\exists xQ(x)}}{P(a),\forall x\,(P(x)\supset Q(x))\to\exists xQ(x)}\qquad \dfrac{\dfrac{Q(b)\to Q(b)}{Q(b)\to\exists xQ(x)}}{Q(b),\forall x\,(P(x)\supset Q(x))\to\exists xQ(x)}}{\dfrac{P(a)\vee Q(b),\ \forall x\,(P(x)\supset Q(x))\to\exists xQ(x)}{\dfrac{(P(a)\vee Q(b))\wedge\forall x\,(P(x)\supset Q(x))\to\exists xQ(x)}{\to(P(a)\vee Q(b))\wedge\forall x\,(P(x)\supset Q(x))\supset\exists xQ(x)}}}$$

Fig.4 A simple shema construction

Then the following formulae is a simple schema.

$$\mathit{schema}_g = (P(a) \vee Q(b)) \wedge \forall x(P(x) \supset Q(x)) \supset \exists x.Q(x)$$

,where P,Q are 2nd order predicate variables. The proof tree for *schema*$_g$ is given by replacing the predicates p, q with P, Q as illustrated in Fig.4.
This means that the type of *schema*$_g$ is the following.

$$[P(a) \rightarrow P(a)] \rightarrow [Q(a) \rightarrow Q(a)] \rightarrow [Q(b) \rightarrow Q(b)] \rightarrow [\rightarrow (P(a) \vee Q(b)) \wedge \forall x\ (P(x) \supset Q(x)) \supset \exists x\ Q(x)]$$

and the proof term of *schema*$_g$ is given as follows:
proof(*schema*$_g$)=imp-R(and-L(or-L(all-L(imp-L(P(a)→P(a),some-R(Q(a)→Q(a))),
thin-L(some-R(Q(b)→Q(b))))))).
From this *schema*$_g$, the proof of any formulae obtained by replacing the symbols P,Q of *schema*$_g$ with any formulas can be derived. This depends on the following well-known property.

[Proposition 3.1:Formulae substitution rule]
If a sequent $\Gamma \rightarrow A$ is provable, then the substituted formulae $\Gamma[P{:=}p(x_1,x_2,\ldots,x_n)] \rightarrow A[P{:=}\ p(x_1,x_2,\ldots,x_n)]$ is also provable, where $[P{:=}p(x_1,x_2,\ldots,x_n)]$ is a substitution of the formulae in the form of $P(t_1,t_2,\ldots,t_n)$ in $\Gamma$ and A with $p(x_1{:=}t_1,x_2{:=}t_2,\ldots,x_n{:=}t_n)$.

Example 3.1 The pıoof of the following formulae h which is obtained from schema$_g$ by substituting P and Q by p ⊃r and s∧ t respectively has similar proof structure to schema$_g$ as shown in Fig.5.
$h=[\rightarrow ((p(a) \supset r(b)] \vee (s(b) \wedge t(b))) \wedge \forall x((p(x) \supset r(x)) \supset ((s(x) \wedge t(x)) \supset \exists x.(s(x) \wedge t(x))]$
Its proof term is obtained as proof(h)=[proof(schema$_g$)](p ⊃ r)(s∧t).

$$\underline{s(a) \wedge t(a) \rightarrow s(a) \wedge t(a)}$$

$$\underline{p(a) \supset r(a) \rightarrow p(a) \supset r(a) \quad s(a) \wedge t(a) \rightarrow \exists x(s(x) \wedge t(x))} \qquad \underline{s(b) \wedge t(b) \rightarrow s(b) \wedge t(b)}$$

$$\underline{(a) \supset r(a)\ ,p(a) \supset r(a) \supset s(a) \wedge t(a) \rightarrow \exists x(s(x) \wedge t(x))} \qquad \underline{s(b) \wedge t(b) \rightarrow \exists x(s(x) \wedge t(x))}$$

$$\underline{p(a) \supset r(a), \forall x(p(x) \supset r(x) \supset s(x) \wedge t(x)) \rightarrow \exists x(s(x) \wedge t(x)) \quad s(b) \wedge t(b), \forall x\ (p(x) \supset r(x) \supset s(x) \wedge t(x)) \rightarrow \exists x(s(x) \wedge t(x))}$$

$$\underline{(p(a) \supset r(a)) \vee (s(b) \wedge t(b))\ ,\ \forall x\ ((p(x) \supset r(x)) \supset s(x) \wedge t(x)) \rightarrow \exists x s(x) \wedge t(x)}$$

$$\underline{(\ (p(a) \supset r(a)) \vee (s(b) \wedge t(b))) \wedge \forall x\ ((p(x) \supset r(x)) \supset s(x) \wedge t(x)) \rightarrow \exists x s(x) \wedge t(x)}$$

$$\rightarrow [(\ p(a) \supset r(a)) \vee (s(b) \wedge t(b))] \wedge \forall x\ (\ p(x) \supset r(x)) \supset (s(x) \wedge t(x))) \supset \exists x.(s(x) \wedge t(x))$$

Fig.5 The proof of h by analogy .

3.2 Schema with constraints

In this section,we shall discuss the relation between the proof terms of schemata with constraints and their instances. Let *form*(g($\dot{A}$)) be a formula whose proof has already been derived as *proof*(g($\dot{A}$)), and let *form*(g($\dot{X}$)) and *proof*(g($\dot{X}$)) be the formula and proof term obtained from *form*(g($\dot{A}$)) and *proof*(g($\dot{A}$)) by replacing some of the symbols in $\dot{A}$ with the symbols in $\dot{X}$ respectively, where $\dot{A}$ and $\dot{X}$ are the list of predicates in *form*(g($\dot{A}$)) and *form*(g($\dot{X}$)). It is noted that *proof*(g($\dot{X}$)) is not always completed. We denote a completed proof of *proof*(g($\dot{X}$)) by *proof**(g($\dot{X}$)). In the followings, we take *form*(g($\dot{X}$)) as a schema constructed from g, and sometimes denote *form*(g($\dot{X}$)) as schema$_g$.For example, let *form*(g(p,q))=[p(a)∨ q(b)]∧∀x(p(x) ⊃ q(x))⊃ ∃x.q(x). Then we have schema$_g$= *form*(g(Φ, Ξ,Ψ, Θ)) = $(\Phi(a) \vee \Psi(b)) \wedge \forall x(\Xi(x) \supset Q(x)) \supset \exists x.\Theta(x)$ as one of the schemata. The proof term of this schema *proof*(g(Φ, Ξ,Ψ,Θ)) is obtained as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{\boxed{\Phi(a) \rightarrow \Xi(a)} \quad \dfrac{\boxed{Q(a) \rightarrow \Theta(a)}}{Q(a) \rightarrow \exists x\Theta(x)}}{\Phi(a), \Xi(a) \supset Q(a) \rightarrow \exists x\Theta(x)}}{\Phi(a), \forall x\,(\Xi(x) \supset Q(x)) \rightarrow \exists x\Theta(x)} \quad \dfrac{\dfrac{\boxed{\Psi(b) \rightarrow \Theta(b)}}{\Psi(b) \rightarrow \exists x\Theta(x)}}{\Psi(b), \forall x\,(\Xi(x) \supset Q(x)) \rightarrow \exists x\Theta(x)}}{\Phi(a) \vee Q(b)\,,\ \forall x\,(\Xi(x) \supset Q(x)) \rightarrow \exists x\Theta(x)}}{\dfrac{(\Phi(a) \vee \Psi(b)) \wedge \forall x\,(\Xi(x) \supset Q(x)) \rightarrow \exists x\Theta(x)}{\rightarrow (\Phi(a) \vee \Psi(b)) \wedge \forall x\,(\Xi(x) \supset Q(x)) \supset \exists x\Theta(x)}}$$

Constraints:
$\Phi(a) \rightarrow \Xi(a)$
$\Psi(b) \rightarrow \Theta(b)$
$Q(a) \rightarrow \Theta(a)$

Fig.6 proof schema.

It is noted that each leaf of *proof*(g($\dot{A}$)) is of the form A, $\Gamma \rightarrow$ A,Δ, and the corresponding sequents of *proof*(g($\dot{X}$)) may be partial proof. That is,the sequents at the leaves of the proof tree are the subproofs which should be proved further. They are called the constraints. In the example of Fig.6, we have to prove the constraints and combine with *proof*(g(Φ,Ξ,Ψ,Θ)) to obtain the complete proof *proof**(g(Φ,Ξ,Ψ,Θ)) from *proof*(g(Φ,Ξ,Ψ,Θ)).

Let constr(g($\dot{X}$:$\dot{A}$)) be the set of constraints between *form*(g($\dot{A}$)) and *form*(g($\dot{X}$)) . For example, the formulae *form*(g(p,q,r,s))=$(p(a) \vee q(a)) \wedge \forall x(r(x) \supset s(x)) \supset \exists x.t(x)$ is provable if the constraints $p(a) \rightarrow s(a)$, $r(b) \rightarrow t(b)$, $r(a) \rightarrow t(b)$ are all provable.This intuitive meaning is given in the following inference rule.

$$\dfrac{\textit{proof}(g(\dot{X})) \quad \textit{proof}(\mathrm{constr}(g(\dot{X}:\dot{A})))}{\textit{proof}^*(g(\dot{X}))}$$

[Theorem 3.2] If *form*(g($\dot{A}$)) is provable, then the proof of schema *form*(g($\dot{X}$)) is provable. The proof *proof**(g($\dot{X}$)) is given by patching the proof(g($\dot{X}$)) with the proof of constraints *proof*(constr(g($\dot{X}$:$\dot{A}$))) which are introduced according to the used inference rules.

4. Schema Construction by Abstraction

A schema is a meta representation for formulas which are syntactically similar, and its proof term represents the proof type. We can consider the proof term of each schema as the specification of proofs. The proofs of the instances of the schema have the similar structure. This means that each instance formulae of a schema is a realization of the specification corresponding to the schema and its proof is an instance of the proof schema. As we have observed in the previous section, the generalization for g,h is performed by transforming them to proof term representations *proof*(g), *proof*(h) using higher order unification algorithm. Concerning to the higher order unification algorithm, the other articles should be referred [7,9,12,15]. We observe this by an example.

Let h be a formula such that

$$h= ((p(a) \wedge r(a)) \vee (q(a) \wedge r(a)) \wedge \forall x(p(x) \supset q(x)) \supset \exists x.q(x).$$

Here, we assume that we want to solve this by the analogy with g.

$$g =[p(a) \vee q(b)] \wedge \forall x(p(x) \supset q(x)) \supset \exists x.q(x).$$

Their proof terms are given in the following forms, where axiom parts of *proof*(h) are arranged according to the proof structure.

g====> *proof*(g)=imp-R(and-L(or-L(all-L(imp-L(p(a)→p(a),some-R(q(a)→q(a)))),
thin-L(some-R(q(b)→q(b))))))).

$$\Uparrow \textit{ generalization}$$
$$\Downarrow$$

h====>*proof*(h)= imp-R(and-L(or-L(all-L((imp-L((p(a)∧r(a))→p(a),some-R(q(a)→q(a)))),
thin-L(some-R((q(a)∧r(a))→q(a))))))).

By the generalization, we get the following proof schema.

imp-R(and-L(or-L(all-L((imp-L(Φ(a)→P(a),some-R(Q(a)→Q(a)))),
thin-L(some-R(Ψ(a)→Q(a))))))).

Then we get the following schema $schema_g$

$$schema_g = (\Phi(a) \vee \Psi(b)) \wedge \forall x(P(x) \supset Q(x)) \supset \exists x.Q(x)$$

constraints: $\Phi(a) \to P(a),\ \Psi(b) \to Q(b)$

, where Φ and Ψ are predicate variables. The *proof*($schema_g$) and proof(constr($\Phi(a) \to P(a), \Psi(b) \to Q(b)$) are obtained as in Fig.7(a),(b).

$$\dfrac{\dfrac{\dfrac{\Phi(a) \to P(a) \qquad \dfrac{Q(a) \to Q(a)}{Q(a) \to \exists x Q(x)}}{\Phi(a), P(a) \supset Q(a) \to \exists x Q(x)}}{\Phi(a), \forall x\,(P(x) \supset Q(x)) \to \exists x Q(x)} \qquad \dfrac{\dfrac{\Psi(b) \to Q(b)}{\Psi(b) \to \exists x Q(x)}}{\Psi(b), \forall x\,(P(x) \supset Q(x)) \to \exists x Q(x)}}{\dfrac{\Phi(a) \vee Q(b), \forall x\,(P(x) \supset Q(x)) \to \exists x Q(x)}{\dfrac{(\Phi(a) \vee \Psi(b)) \wedge \forall x\,(P(x) \supset Q(x)) \to \exists x Q(x)}{\to (\Phi(a) \vee \Psi(b)) \wedge \forall x\,(P(x) \supset Q(x)) \supset \exists x Q(x)}}}$$

(a) Proofs for $schema_g$

$$\dfrac{\text{Constraints:}}{\begin{array}{c}\Phi(a) \to P(a) \\ \Psi(b) \to Q(b)\end{array}} \qquad \dfrac{\dfrac{P(a) \to P(a)}{P(a), R(a) \to P(a)}}{P(a) \wedge R(a) \to P(a)} \qquad \dfrac{\dfrac{Q(a) \to Q(a)}{Q(a), R(a) \to Q(a)}}{Q(a) \wedge R(a) \to Q(a)}$$

(b) Proofs for constraints.

Fig.7 Schema with constraints for proving h.

5.Proving by Analogy

The rough sketch of this procedure is given as follows. We assume that standard schema have already been obtained as schema database, and let its elements be $S_1, S_2, \ldots, S_n$.Firstly, a given target problem w is checked if some similar guiding problem exists or not. There are two cases for this step. One is to construct a schema from g and w by generalization. The other is to search a schema on the schema database which is unifiable both with g and w. We are intending to develop a system which combine 'ie both cases. This similarity check is examined using 2nd order matching algorithm. If there exists a schema S which match to w, then a unifier is produced. The proof of w is derived by σ(proof(S))+proof(constraints).

Procedure

```
 input: w (formulae) ;output: proof(w);
 begin
   Find a schema S which match with w
       (1) if there is no such schema then stop and output "prove by yourself"
       (2) else choose (in nondetermistic ) a schema S;
              (2-1)compute unifier such that $\sigma$(S)= w
              (2-2) check if it satisfies the constraints
                     if it satisfies then output $\sigma$[proof(S)]+proof(constraint)
                               else "prove by yourself"
 end
```

It is noted that the procedure uses only 2nd order matching, it is realized in the computable scope.

[Theorem 4.1] The analogical proof reasoning procedure proposed here for LK is computable.

In the previous example in Section 4, we get a substitution $\sigma=\{\Phi:=p\wedge r,\ \Psi:=q\wedge r, P:=p, Q:=q\}$ by the matching of h with *schema*$_g$ as typed terms, Then the proof(h) is derived as

$\mathit{proof}(h)=[\lambda\Phi\lambda\Psi\lambda P\lambda Q.\mathit{proof}(\mathit{schema}_g)](p\wedge r)(q\wedge r)(p)(r)+[\lambda\Phi\lambda\Psi\lambda P\lambda Q.\mathit{proof}(\mathrm{constr}(\Phi\rightarrow P,\Psi\rightarrow Q))]\ (p\wedge r)(q\wedge r)(p)(r)$

6.Discussions

We proposed an analogical reasoning for LK proof system based on higher-order abstraction. By this approach, a kind of proof system by analogy can be realized in natural way. Especially, it holds a similar interpretation of the analogy to the formulae as type concept such that the schemata corresponds to specifications and object proofs to their realizations. The procedure proposed here can be realized using the higher order unification algorithm for typed terms in the computable scope.

However, there exist several important problems to be solved . The most essential one is to design an efficient unification algorithm.The other problem is that the schema expressed by second order variables are too general for many cases. Hence undesirable unifiers will be output sometimes. In order to specify the schema more precisely, some additional axioms should be attached to such schema. Further,the obtained proof by this method is not always good. To translate the obtained proof to a better proof form is one of the interesting problems concerning to this topic.

Acknowledgement
This research was supported partly by the Telecommunications and Advancement Foundation.

7. References

[1]P.B.Andrew: An Introduction to Mathematical Logic and Type Theory, Academic Press.inc.,1986.

[2] B.Brock,S.Cooper and W.Pierce: Analogical Reasoning and Proof Discovery, LNCS, No.310, 9th Int.Conf on Automated Deduction, pp454 ~468,1988

[3]M.R.Donat, L.A.Wallen: Learning and Applying Generalized Solutions using Higher Order Resolution, LNCS,No.310,Int.Conf on Automated Deduction,pp41~60,1988.

[4]K.Fujita,M.Harao: Proving Based on Similarity, Proc. of 2nd Workshop on Algorithmic Learning Theory, Japan Society of Artificial Intelligence, pp.213-223,1991,10

[5] R.Greiner: Abstraction-Based Analogical Inference, Herman (Ed.), Analogical Reasoning,1988,pp.147-170.

[6]M.Haraguchi,S.Arikawa: A Formulation of Analogical Reasoning and Its Realization, Journal of JSAI, Vol.1 No.1,pp132~139,1986.

[7] M.Harao,K.Iwanuma: Complexity of Higher-Order Unification Algorithm, Computer Software, Vol.8,No.1, Jan.1991,pp.41-53

[8]M.Harao: Knowledge Processing Based on Higher-Order Unification, Technical Report JSAI, SIG-FAI-8903-3, pp21~30,1989.

[9]G.P.Huet, B.Lang: Proving and Applying Program Transformations Expressed with Second- Order Patterns, Acta Informatica, 11, pp 31 ~55, 1978.

[10]S.Kedar-Cabelli: Analogy from a unified perspective, Herman (Ed.), Analogical Reasoning,1988,pp.65-103.

[11] D.A.Miller, A.Felty: An Integration of Resolution and Natural Deduction Theorem Proving, Proc. of AAAI86,pp.198-202.1986

[12]S.Muggleton, C.Feng: Efficient Induction of Logic program, Proc. of Workshop on Algorithmic Learning Theory, 1990,10,pp.368-382.

[13]I.Niniluoto: Analogy and Similarity in Scientific Reasoning, Herman(Ed.) Analogical Reasoning, Kluwer Academic Pub. 1988, pp.271-298.

[14] D.A.Plaisted: Theorem Proving with Abstraction, Artificial Intelligence, Vol.16,No.1, 1981,pp.47-108.

[15] W.Snyder, J.Gallier: Higher Order Unification Reviced, Journal of Symbolic Computation,No.8,1989,pp.101-140.

[16] T.B. de la Tour, R.Caferra: Proof Analogy In Interactive Theorem Proving, IJCAI '87, pp95~99,1987.