Knowledge Acquisition by an Intelligent Acting Agent

Michael Kandefer and Stuart C. Shapiro

Department of Computer Science and Engineering, Center for Cognitive Science,
and The National Center for Multisource Information Fusion
University at Buffalo, The State University of New York
{mwk3,shapiro}@buffalo.edu

Abstract

We present a solution to McCarthy's Second Telephone Number Problem. This problem requires an agent to: realize that it lacks some knowledge to complete a task; know the external knowledge sources it can use to obtain the knowledge; know how to obtain the missing knowledge from those sources; actually obtain the missing knowledge; and use the obtained knowledge to complete the task. We explain how a SNePS-based agent solves the problem, and the features of SNePS and the GLAIR architecture that facilitate the solution. These features include: the integration of reasoning and acting; the ability to reason about actions; and the ability to represent and reason about the knowledge contained in external knowledge sources. This latter feature is accomplished through the process of grounding of symbols of the knowledge representation in symbols of an external language.

Introduction

McCarthy (1979) presents several problems, the solutions of which would require a common-sense-reasoning agent to use concepts to remove the need for modification of the first-order logic representations. The goal of this paper is to detail a solution for one of these problems, which we call "McCarthy's second telephone number problem." McCarthy's statement of this problem is:

A computer program that wants to telephone someone must reason about who knows the number. More generally, it must reason about what actions will obtain needed knowledge. Knowledge in books and computer files must be treated in a parallel way to knowledge held by persons. [Furthermore, a] program must often determine that it does not know something or that someone else doesn't. (McCarthy 1979)

The problem requires of an agent five abilities:

1. Determining the absence of the requisite knowledge for completing the task at hand. In the example problem this information is an individual's telephone number. The task is to call that individual.

2. Determining the external information sources that might contain the required information. A telephone book or other agents are examples.

3. Knowing how to consult the source(s).

4. Retrieving and representing the needed information.

5. Completing the task with the acquired information.

Although this telephone number problem was first posed 28 years ago, we know of no paper that presents a solution to it, nor does McCarthy[1]. However, this is not to say no research has been done on the five processes discussed above, nor on similar problems[2]. Certainly McCarthy (1979) addresses similar problems throughout his paper. However, unlike those problems this one requires of the agent an embodiment and the ability to "transfer" information from the external environment into the agent's belief-space.

This paper examines the five processes involved and reports a solution to McCarthy's second telephone number problem using a simulated environment in which the agent can perform the action of calling individuals by pressing buttons on a simulated telephone. The solution is implemented in SNePS (Shapiro 2004), a knowledge representation, reasoning, and acting system; and uses the GLAIR agent architecture(Hexmoor, Lammens, & Shapiro 1993). Both contain helpful and necessary features for commonsense reasoning tasks, such as:

- the integration of reasoning and acting,

- the ability to reason about actions,

- the ability to reason about knowledge contained in external information sources by grounding internal symbols in the agent's "language of thought" in symbols of an external language,

- the ability to reason about its own knowledge.

GLAIR

The agent designed to solve the second telephone number problem uses the Grounded Layered Architecture for Integrated Reasoning (GLAIR). This architecture is split into three layers:

---

[1]Personal communication.

[2]Such as Moore's work on the knowledge preconditions for a planning problem using a safe combination written on a scrap of paper (Moore 1980).

The Knowledge Layer (KL) contains what is required for conscious reasoning. The agent's propositional belief base, high-level plans, and reasoning system are all contained in this level.

The Perceptuo-Motor Layer (PML) serves three primary functions: (1. PMLa) Defining the primitive actions available to the agent (2. PMLb) Establishing a communication layer between PMLa and PMLc and (3. PMLc) Interacting with the agent's embodiment—its sensors and actuators.

The Sensory Actuator Layer (SAL) and Environment contains the implementation of the agent's embodiment (hardware or software simulation) and, if needed, the implementation of a simulated environment.

The following sections detail the SNePS/GLAIR solution to the McCarthy telephone problem.

Solution Details

Representing Requisite Knowledge

Before an agent can determine if it is missing the knowledge needed to call some individual it needs a method of storing beliefs about an individual's telephone number. This is a KL task.

A SNePS-based agent has a first-person belief base, meaning the propositions contained in it are justified through the agent's reasoning and are not necessarily true in the world. Moreover, the agent's internal belief structure is a set of propositions in the agent's personal language of thought, which is independent of the natural language used to express these beliefs (Shapiro 1993) to the user. SNePSLOG, a logical language resembling higher-order logic that can be used as an interface to SNePS, was used to construct our agent and is used in this paper.

The two SNePSLOG expressions important for our solution are the first-order representations for telephone numbers (to be discussed below) and the belief that some agent has a telephone number. Possession of anything is represented by the predicate *Has(p,r,o)* which means that: "agent *p* possesses entity *o* as an *r*" or "*p*'s *r* is *o*". Both the possessor and the object of possession are denoted by SNePSLOG terms that might either be created when the belief base is first initialized, or when the agent interacts with the world. Additional proposition-valued terms used by the agent are:

- *Name(e,pn)* - The proper-name of entity *e* is *pn*.

- *Value(e,v)* - The value of entity *e* is *v*.

For example, *Name(b5, Mikelex)* is the representation of our agent's belief that the proper-name of the entity denoted by *b5* is the entity denoted by *Mikelex*. The individual constant *Mikelex* denotes the lexeme "Mike". How this denotational connection is made will be detailed in a later section.

Similarly, the lexeme-denoting terms *n0, n1, ..., n8,* and *n9* are used for the digits in telephone numbers. Thus, the representation of the proposition, "Mike's telephone number is '555-5612' " is

```
Has(b5,
 compCat(lex(telephonelex),lex(numberlex)),
 b10).
Name(b5, Mikelex).
Value(b10,
```

```
N(n5,N(n5,N(n5,N(n5,N(n6,N(n1,n2)))))))).
```

In the above, *lex(w)* is a term denoting the entity that can be expressed by the lexeme denoted by *w*. The *compCat(m,c)* expression denotes the complex category which is a subcategory of the category denoted by *c*, and is modified by *m*. Besides *compCat(m,c)* being a subcategory of *c*, the precise relation between the two is left ambiguous. *N(d,n)* is a term denoting the digit sequence that begins with the digit denoted by *d*, and continues with the digit or digit sequence denoted by *n*. The telephone-calling agent can infer that *N(d,n)* denotes a digit sequence by use of the recursive SNePSLOG rule shown in Fig. 1. *N(d,n)* terms can be used by the agent to dial a telephone number one button press at a time, or to denote a telephone number.

SNeRE Basics

SNeRE is an online acting system, meaning the agent reasons about the state of the world while it acts. These acts can either be mental acts or acts requiring an agent embodiment, but in either case the acts are parts of the agent's belief base and acted upon using the SNePS reasoning system. SNeRE interprets a number of SNePSLOG expressions[3] to effect the agent's actions. The following are used by this agent:

- **Proposition-forming functions:**

  - *ActPlan(a1,a2)*: The plan for performing the defined act *a1* is to perform act *a2*.

- **SNeRE primitive acts**

  - *snsequence(a1,a2)*: Perform act *a1* and then act *a2*. For action sequences of $n > 2$ actions, *snsequencen(a1,a2,...,an)* is used.

  - *snif({if(p1,a),...,if(pn,an)[,else(da)]})*: Using backward inference, determine which of the *pi* hold. If any do, randomly choose one of them, say *pj*, and perform *aj*. If none of the *pi* can be inferred, and if *else(da)* is included, perform *da*. Otherwise, do nothing.

  - *withsome(x, p(x), a(x), da)*: Using backward inference, determine which, if any, entities satisfy the open proposition *p(x)*. If any do, randomly choose one, say *e*, and perform *a(e)*. If no entity satisfies *p(x)*, perform *da*. *withall* is like *withsome*, but performs *a(e)* on all *e* that satisfy *p(x)*.

- **SNeRE Mental Acts**:

  - *disbelieve(p)*: If *p* is a believed proposition, it is no longer believed.[4]

  - *believe(p)*: Proposition *p* is believed as a hypothesis, and forward inference is done with *p*. This can cause new propositions to be believed. Additionally, a limited form of belief revision is performed which will disbelieve $\sim p$ if necessary.

---

[3]For complete documentation of the SNeRE constructs, see (Shapiro 2004)

[4]This does not cause the agent to believe $\sim p$.

```
all(d)(Member(d, lex(digitlex))
      => all(n)({Member(n,lex(digitlex)),Member(n,compCat(lex(digitlex),lex(sequencelex)))}
                v=> Member(N(d,n), compCat(lex(digitlex),lex(sequencelex))))).
```

Figure 1: Recursive SNePSLOG Rule that establishes what entities are members of the category of digit sequences.

Introspection on the Belief Base

The agent requires a form of mental introspection to determine if it holds a belief or not, and a method of acting on the results of that introspection. Since this introspection is a conscious mental action some method for representing, and executing these acts is required. For our purposes SNeRE constructs *withsome* and *ActPlan* are used for the tasks of belief base introspection and acting on this introspection.

For example, using the above expressions our agent can determine if it knows the value (*n*) of Mike's (*b5*) telephone number and dial it using the following SNePSLOG expression:

```
withsome({n,n1},
 (Has(b5,compCat(lex(telephonelex),
                 lex(numberlex)),n1)
  and Value(n1,n)),
 dial(b5,n), lookup(b5))
```

A generalization of the above example is used by the agent in its *dial-person* defined act (specified in Fig. 2b) to determine if the agent can place the call to some individual *p*[5] with telephone number value *n* using the defined act *dial(p,n)*. However, if the agent doesn't know the telephone number it needs to consult an appropriate information source, if available, using *lookup*. The plans for these defined acts will be discussed in the following sections.

Reasoning about External Information Sources

In order to reason about which external sources need to be searched the agent employs the use of the *lookup* defined act, but this procedure relies on some initial set up work done at the beginning of the plan for *act(lex(calllex),p)* (Fig. 2a). Here the term *act(a,o)* denotes the act whose action is *a* and whose object is *o*. So the act *act(lex(calllex),p)* denotes the act of calling person *p*. Similarly, *act2* is used for an action of arity two. When performing the call procedure, the agent first believes that it has not yet searched any of the information sources it knows of, nor consulted the user for the telephone number of the person being called. To do this the agent uses the SNeRE constructs *withall*, *snsequence*, and *believe*; along with the negation of the SNePSLOG terms *Searched* and *ConsultedUser*, the former of which is applied to all those entities the agent believes to be information sources (*compCat(lex(informationlex),lex(sourcelex))*).

Next, the agent picks up the phone and retrieves the number using *dial-person*, which may result in the execution of *lookup*, if the agent doesn't know the telephone number.

lookup (Fig. 2c), like *dial-person*, utilizes *withsome* on the conjunction of three propositions, and is used if the

agent believes there exists some information source *i* that it hasn't yet searched such that *i* contains *p's* telephone number. If such an information source is known to the agent, it will search it for *p's* telephone number using the *search* primitive-action, denoted as *act2(lex(searchlex),p,i)*, and then recursively try *dial-person(p)* again. Provided the agent has found the number it will dial it at this point, but if not, it will repeat the procedure until there are no more sources to consult, including the user of the system.

This procedure allows the agent to determine those information sources it should check for a person's telephone number through the use of the *Contains(o,t,e)* expression, which represents the proposition, "Object [o] contains [t] type information on entity [e]". The agent believes proposition-valued terms of the above expression through the use of two SNePSLOG rules depicted in Fig. 3. The two rules state that if the agent believes a person is a student or professor, then their telephone number is contained in the information source named 'Campus Database' (denoted by *bCD*), and if a person is not a student nor a professor, then their telephone number is contained in the information source named 'Phonebook' (denoted by *PbB*). Admittedly the rules are a bit naïve, but they are an example of how the agent can reason about what type of information an external information source contains, and allows the agent to determine which source to choose when searching for an someone's telephone number in the plan for 'call'.

Interacting and Symbol Anchoring with External Sources

In order for the agent to complete the task, a method of obtaining information from external sources is required. As such, this agent requires some sort of embodiment that can interact with these sources and "transfer" the information from them to its internal belief base. This transfer occurs through the use of the PML of GLAIR.

The PML serves two primary connection tasks between the KL and the agent's embodiment in the world (SAL), which is accomplished by linking KL terms to PML symbols via a mapping procedure we call "alignment" (Shapiro & Ismail 2003). The first is defining the primitive actions and connecting them with the agent's embodiment, the second is establishing a language facility that translates between the agent's internal representations to English text and vice versa.

Primitive Acts and Embodiment Both the agent's embodiment and the PML are implemented in Lisp, as such there is no need to establish any communication layer other than Lisp function calls. The primitive actions available to the agent are as follows:

---

[5]This and other plan specifications only apply to individuals that the agent believes to be people. Though not illustrated here, this type of rule can be used to specify different plans to perform the same act on different categories of objects.

```
;;; a. Top level 'call' procedure
all(p)(Member(p,lex(personlex))
   => ActPlan(act(lex(calllex),p),
        snsequence4(withall({i,t},
                            Member(i,compCat(lex(informationlex),lex(sourcelex))),
                            believe(~Searched(i)),  do(nothing)),
                    believe(~ConsultedUser(p)), do(lex(pickuplex)), dial-person(p)))).
;;; b. dial-person procedure
all(p)(Member(p,lex(personlex))
   => ActPlan(dial-person(p),
        withsome({n,n1},
                (Has(p,compCat(lex(telephonelex),lex(numberlex)),n1) and Value(n1,n)),
                dial(p,n),
                lookup(p)))).
;;; c. lookup procedure
all(p)(Member(p,lex(personlex))
   => ActPlan(lookup(p),
        withsome(i, (Member(i,compCat(lex(informationlex),lex(sourcelex))) and
                    Contains(i,compCat(lex(telephonelex),lex(numberlex)),p) and
                        ~Searched(i)),
                snsequence3(act2(lex(searchlex),p,i),
                        believe(Searched(i)),
                        dial-person(p)),
                snif({if(~ConsultedUser(p),
                        snsequence3(act(lex(asklex),p),
                                believe(ConsultedUser(p)),
                                dial-person(p)))})))).
```

Figure 2: SNePSLOG specification for the high level act of calling someone, split between three procedures: *call*, *dial-person*, and *lookup*.

- **Information retrieving primitives**
  - **search(*p*, *s*)** - The agent looks for all entries related to person *p* in information source *s*. If an entry is found with *p*'s name and a telephone number, the agent believes that *p* has this value as their telephone number.
  - **ask-user(*p*)** - Ask the user of the system to enter the phone number for person *p*, if they know it.
- **Dialing Primitives**
  - **pickup** - The agent picks up the phone receiver, turning it on.
  - **putdown** - The agent puts down the phone receiver, turning it off.
  - **press(*n*)** - The agent presses the button labeled *n* on the phone. If the call goes through at this point the agent will hear someone talking on the other end and subsequently believes that it is talking to the person it called.

These primitive acts are used by the KL to control the agent's embodiment in the world. Each KL term of the form *lex(x)* is aligned with the PML primitive action function via the *attach-primaction* function. An example of attaching the *lex(searchlex)* KL term with the *search* primitive action definition is:

```
(attach-primaction
   (build lex ''searchlex'') search)
```

The world is implemented as a Lisp simulation that provides the agent with a simulated telephone and external in-formation sources implemented as text files that contain information entries. The entries as well as the content type vary between the files. In the current implementation two information sources are available, a campus telephone directory listing and a public telephone network listing. The campus listing in a text file contains an individual's name, the building their office is in, their office number and telephone number, for example:

```
Bill NSC 678 555-8989
```

Each entry in the telephone directory listing in a text file contains an individual's name, address, and telephone number, for example:

```
John 305 Pine Tree Lane 479-2344
```

In the case of the agent's *search* and *ask-user* primitive actions, the acts parse the information observed and use the agent's language facility to make assertions into the knowledge base about the individual's telephone number.

Language Facility In order to create a natural language interface for the agent and to provide a means of converting between the agent's internal belief structure, or "language of thought", and natural language, a combination of Lisp code and a Generalized Augmented Transition Network (GATN) grammar was used (a process demonstrated by Shapiro(1982; 1998)). The GATN parses English input in the form of commands, statements, or questions, each of which goes through the following I/O loop:

1. Analysis of the input using the syntax and semantics spec-

ified in the lexicon and grammar along with the the agent's current beliefs.

2. If necessary, the creation of new terms in the knowledge base.

3.(a) If a statement, the agent generates an English sentence from the SNePS proposition asserted (in step 2) by the original statement preceded by the phrase, "I understand that".

 (b) If a question, the answer(s) to the question are inferred from the agent's belief space, and stated in English.

 (c) If a command, the agent attempts to perform the action specified by the command,which can be either a primitive action or a high level act specified in SNePS.

The I/O will not succeed in circumstances where the agent doesn't know: a particular word or grammatical structure; the information requested in a question; or how to perform a particular action specified in a command.

The lexicon is a listing of English lexemes initially known to the agent. Individual telephone numbers, of course, are not all listed in the lexicon. Instead, the morphological analyzer was modified to use ACL Lisp's regular expression package to recognize any sequence of three digits, followed by a hyphen, followed by four digits as a telephone number[6] with the lexical category `value`. The lexeme-denoting terms of the KL are aligned with these lexemes allowing the agent to translate between lexeme and term.

With the lexicon and grammar in hand the agent can then translate between its first-order language of thought and English. For example, the telephone number string "555-1234" becomes the SNePSLOG expression $N(n5,N(n5,N(n5,N(n1,N(n2,N(n3,n4))))))$ when the agent observes the string in the external sources or from the user's input. The opposite translation occurs when the agent is asked to express its knowledge in English. We believe that allowing propositions to be first-class objects in the domain, using proposition-denoting functional terms (Shapiro 1993) and using alignment to connect lexeme-denoting terms with the actual lexemes obviate the need for a quotation or string operator, as describe, for example, in (Davis 1990).

Using the Information

Regardless of the method used by the agent to obtain the telephone number of the person, the agent uses the *dial* defined act (Fig. 4) to call the number. The plan for *dial* is recursive, and split among three plans. Most work is done by *ordered-press*, which works on digit sequences. The first step is to check if the first and second arguments of the digit sequence are digits, if so, then they are pressed using the *press* primitive action in order, and the recursion stops. If not, only the first is pressed and a recursive call is made on the second argument, which is a digit sequence. It should be noted that this defined act illustrates two features of SNePSLOG. The first is that SNePSLOG uses unique variable binding (Shapiro 1986) meaning if one uses *v1* and *v2* as

---

[6]This ignores complications such as area codes, long-distance calls, etc., issues that are ultimately in the realm of message understanding.

---

variables, they will be bound to different entities. The second has been discussed already, this being that SNePSLOG reasoning rules can be used to specify an defined act for different types of entities. In this case, the defined act *ordered-press* is specified for digit sequences composed of two of the same digits, as well as those that don't or are composed of a digit and a nested digit sequence. Like previous tasks, this one is largely a KL task, but uses primitive actions that involve the PML.

Sample Run

Below three sample runs are given. The first demonstrates a request to call someone the agent knows the phone number for, the second someone the agent doesn't and can look up in one of the information sources, and the third illustrates asking the user for the phone number. User input in shown in **bold**. The agent starts with the belief that the distinct individuals named 'Mike', 'Stu', and 'Albert' are people. The agent also initially believes that the individual named 'Stu' is a professor, and that the individual named Albert is neither a student nor a professor. The user's questions are to show the reader the state of the agent's belief base at various times.

First run:
: **What is Mike's telephone number?**
Mike's telephone number is 555-5612.

: **Call Mike.**
I am pressing 5. I am pressing 5.I am pressing 5. I am pressing 5. I am pressing 6. I am pressing 1. I am pressing 2.

Second run:
: **What is Stu's telephone number?**
I don't know.

: **Call Stu.**
Checking Campus Database for Stu's telephone number...
I am pressing 5. I am pressing 5. I am pressing 5. I am pressing 7. I am pressing 8. I am pressing 9. I am pressing 0.
: **What is Stu's telephone number?**
Stu's telephone number is 555-7890.

Third run:
: **What is Albert's telephone number?**
I don't know.

: **Call Albert.**
Checking Phonebook for Albert's telephone number...
I could not find Albert's phone number in any external information source available to me.
Do you know Albert's number? **Yes**
What is Albert's number (e.g. 555-5555)? **555-1234**
I am pressing 5. I am pressing 5. I am pressing 5. I am pressing 1. I am pressing 2. I am pressing 3. I am pressing 4.
: **What is Albert's telephone number?**
Albert's telephone number is 555-1234.

```
all(p)(Member(p, lex(personlex))
        => ({Member(p,lex(professorlex)), Member(p,lex(studentlex))}
            v=> Contains(bDB,compCat(lex(telephonelex),lex(numberlex)),p))).

all(p)(Member(p, lex(personlex))
        => ({~Member(p,lex(professorlex)), ~Member(p,lex(studentlex))}
            &=> Contains(bPB,compCat(lex(telephonelex),lex(numberlex)),p))).
```

Figure 3: The SNePSLOG rules that specify which information source contains which telephone numbers.

```
all(a,v,pn)({Member(a,lex(personlex)), Value(v,pn),
             Has(a,compCat(lex(telephonelex), lex(numberlex)), v)}
            &=> ActPlan(dial(a,pn), ordered-press(pn))).

all(n1,n2)(Member(N(n1,n2),compCat(lex(digitlex),lex(sequencelex)))
    => ActPlan(ordered-press(N(n1,n2)),
         snif({if(Member(n2,lex(digitlex)),
                 snsequence(act(lex(presslex),n1), act(lex(presslex),n2))),
               else(snsequence(act(lex(presslex),n1), ordered-press(n2)))}))).

all(n)(Member(N(n,n),compCat(lex(digitlex),lex(sequencelex)))
    => ActPlan(ordered-press(N(n,n)),
         snsequence(act(lex(presslex),n), act(lex(presslex),n)))).
```

Figure 4: SNePSLOG specification for the high level act of dialing an individual's number.

Conclusions

We have demonstrated a solution to McCarthy's second telephone number problem using the SNePS knowledge representation, reasoning, and acting system. The agent performed the necessary tasks to solve this problem:

1. Determining the absence of requisite knowledge using the *withsome* action,

2. Determining the external information sources that might contain the missing information using a SNePSLOG knowledge-base and rules of inference,

3. Consulting these sources through using a simulated embodiment provided by the GLAIR architecture,

4. Retrieving and representing the needed information by anchoring the symbols of the knowledge base in those of an external communication language using the GLAIR architecture and language facility, in order to represent telephone numbers in both the agent's language of thought and English,

5. Completing the task with the acquired information using the SNePS acting system with integrated reasoning.

References

Davis, E. 1990. *Representations of commonsense knowledge*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Hexmoor, H.; Lammens, J.; and Shapiro, S. C. 1993. Embodiment in GLAIR: a grounded layered architecture with integrated reasoning for autonomous agents. In Dankel II, D. D., and Stewman, J., eds., *Proceedings of The Sixth Florida AI Research Symposium (FLAIRS 93)*, 325-329. Florida AI Research Society.

McCarthy, J. 1979. First order theories of individual concepts and propositions. In Hayes, J.; Michie, D.; and Mikulich, L., eds., *Machine Intelligence 9*. Ellis Horwood Ltd. 129-147.

Moore, R. C. 1980. Reasoning about knowledge, and action. Technical Report 191, Artificial Intelligence Center, SRI International.

Shapiro, S. C., and Ismail, H. O. 2003. Anchoring in a grounded layered architecture with integrated reasoning. *Robotics and Autonomous Systems* 43(2–3):97–108.

Shapiro, S. C. 1982. Generalized augmented transition network grammars for generation from semantic networks. *The American Journal of Computational Linguistics* 8(1):12-25.

Shapiro, S. C. 1986. Symmetric relations, intensional individuals, and variable binding. *Proceedings of the IEEE* 74(10):1354-1363.

Shapiro, S. C. 1993. Belief spaces as sets of propositions. *Journal of Experimental and Theoretical Artificial Intelligence (JETAI)* 5(2&3):225–235.

Shapiro, S. C. 1998. Embodied Cassie. In *Cognitive Robotics: Papers from the 1998 AAAI Fall Symposium, Technical Report FS-98-02*, 156-153. Menlo Park, CA: AAAI Press.

Shapiro, S. C. 2004. *SNePS 2.6.1 User's Manual*. Department of Computer Science and Engineering, State University of New York at Buffalo.