17 Metareasoning as an Integral Part of Commonsense and Autocognitive Reasoning

Fabrizio Morbini and Lenhart Schubert

In this chapter we summarize our progress toward building a self-aware agent based on the notion of explicit self-awareness[1] (Schubert, 2005). An explicitly self-aware agent is characterized by (1) being based on extensive, human-like knowledge about the world and itself and the ability to reason with that knowledge, (2) relying on a transparent (easily understood, and semantically well-defined) knowledge representation, and (3) being able to explain itself and display its self-awareness through natural language dialogues. The second point is not strictly related to self-awareness, but it facilitates implementation of some aspects, such as answer explanation, and simplifies testing and debugging of the agent. In addition, we emphasize the importance of metalevel reasoning in commonsense reasoning and self-awareness, while questioning the common view of agent control structure in terms of separate object-level and metalevel strata. Instead, we suggest a "continual planning" (and execution) control structure wherein the agent's metalevel and object-level reasoning steps mingle seamlessly.

We first review the requirements imposed by explicit self-awareness and by this intermingling of object-level and metalevel reasoning on the knowledge representation and reasoning system and then describe how these have been realized in the new version of the EPILOG[2] system. Then we demonstrate our agent by looking at a few questions, each related to some aspects of self-awareness. Finally, we conclude with a discussion of our long-term plans.

1. Conditions 1 and 3 for explicit self-awareness, by requiring human-like abilities, are intended to set aside weaker notions of self-awareness such as monitoring and regulation of internal parameters (as in operating systems or thermostats), internal event-logging (for example, of function calls and their local environments), or goal-directed behavior (as in heat-seeking missiles or ant colonies). Certainly the conditions are not intended to be *necessary* for all reasonable notions of self-awareness.

2. EPILOG is a reasoning system for Episodic Logic (see Schubert & Hwang, 2000). Its main components are a general inference engine based on backward and forward chaining and a set of specialists to aid the general inference engine in particular types of inferences (e.g., temporal, introspective).

Metareasoning

Schubert (2005) lists a series of requirements on a knowledge representation and reasoning system to enable explicit self-awareness as defined in the previous section. We first summarize these requirements, and then describe in some detail how two of these requirements have been implemented in EPILOG.

1. *Logic framework*: we require an explicit, transparent[3] representation for the agent's commonsense knowledge and metaknowledge that allows for easy browsing/editing/debugging and manipulation by general inference methods. We chose Episodic Logic (EL) as an extension of first-order logic (FOL) that is more fully adapted to the expressive devices of natural language (NL) (e.g., see Schubert and Hwang, 2000).
2. *Events and situations*: the agent must be able to refer to events described by complex sentences (e.g., the event described by EPILOG's failure to answer a question). This capability has always been an integral part of EL and therefore of EPILOG.
3. *Generic knowledge*: much of everyday knowledge is expressed using generic or probabilistic adverbs such as *usually* or *probably*. In EPILOG generics are expressed using probabilities, though this support is very limited and in need of further development.
4. *Attitudes and autoepistemic inference*: the ability to reason about one's own knowledge and to represent one's beliefs is fundamental to self-awareness. Our commitment is to a formal computational notion of knowing as described by Kaplan (2000) and Kaplan and Schubert (2000). We describe below how the basic machinery needed for this has been implemented in EPILOG.
5. *Metasyntactic devices*: a self-aware agent needs to be able to refer in its logical language to the syntax of that language itself. How some of these devices have been implemented in EPILOG will be described later in this section.

The following subsections focus on how the last two requirements, those most central to self-awareness and metareasoning, have been implemented in EPILOG. As a notational alert, we should mention that EL uses infix notation for predication and prefix notation for function application; e.g., (*EL* (*very Expressive*)) states that *EL* is very expressive; the predicate (*very Expressive*) is infixed, while the predicate modifier *very* (a function that maps monadic predicates to monadic predicates) is prefixed.

Substitutional Quantification and Quasi-Quotation

As motivated and exemplified by Schubert (2005) and Morbini and Schubert (2007), it is important to be able to refer to syntactic elements of EL in an EL formula itself.

3. An example of knowledge that is not explicit and transparent would be the matrices of numbers learned by a machine-learning technique.

For instance, this allows a formal treatment of axiom schemas, enables EPILOG to classify its own predicates and formulas, and allows EPILOG to choose executable procedures for solving certain kinds of problems deliberately, based on axioms about their effects.

To enable this kind of "syntactic self-awareness," EPILOG currently supports two devices: substitutional quantifiers and quasi-quotation. Their implementation is straightforward and posed no major problems. The two main modifications required concerned the following components:

1. *EL-Parser*: we added substitutional quantifiers and metavariables. A metavariable is a particular type of variable that is bound by a substitutional quantifier. A substitutional quantifier is much like a normal quantifier except that it quantifies over substitutions of expressions of a particular category for the metavariable. For example, $(\forall_{wff}\ w\ (w \Rightarrow (me\ Know\ (that\ w))))$[4] quantifies over substitutions of EL well-formed formulas for the variable w. The truth conditions of the formula are that all instances of the formula are true under the object-level semantics, when EL well-formed formulas (not containing metavariables) are substituted for w.
2. *Quasi-quotation*: written with a quote sign (apostrophe), quasi-quotation accepts as argument an expression that may contain metavariables; substitution for such metavariables treats quasi-quotes as transparent. As an example of the use of quasi-quotation, we can express in EL that the "=" predicate is commutative by writing $('= (Commutative\ ELpredicate))$ where *Commutative* is a predicate modifier and *ELpredicate* is a predicate (true of certain syntactic objects). Other examples that will be seen later are the use of *AppearancePred* and *AppearanceFactAbout* to describe internal predicates and formulas.
3. *Unification routines*: since we have provided for metavariables, unification should be able to unify not only variables with terms but also metavariables with EL expressions of the appropriate categories. For example, we can unify $(me\ Know\ (that\ w))$ with $(me\ Know\ (that\ (?x\ Foo)))$, yielding unifier $\{(?x\ Foo)/w\}$.

Recursive QA

Morbini and Schubert (2007) described the basic properties that a computational notion of knowing should have. In that work, we indicated why *knowing* is very different from *being able to infer*, and we referred to Kaplan and Schubert's algorithmic ASK mechanism as a basis for *knowing*. As was shown by Kaplan and Schubert (2000), certain axiomatic constraints on ASK can assure soundness of simulative inference, and we can also assure compliance with certain natural requirements on knowing, such as that $(\phi\ and\ \psi)$ is known iff ϕ is known and ψ is known. Here we describe how ASK is supported in EPILOG. The intuitive way to implement ASK (and thus to answer

4. Of course, the omniscience claim expressed by this formula is absurd.

questions that involve predicates like *Know* or *Believe*) is to allow for question-asking within a question-answering (QA) process, where the subordinate QA process is guaranteed to terminate relatively quickly. If a question requires prolonged reasoning, the answer is by definition not known. Answering questions about nested beliefs thus involves further nesting of QA processes. That is the basic idea behind recursive QA.

Again, implementing this is straightforward in any system with a clean and modular implementation of the QA process. What is needed is that the QA process must work only on local variables, and the same must be true for all systems on which QA depends (e.g., knowledge base, unification, inference and normalization).

In addition to having a modular system, one needs a way to connect inference with the QA process so that this process can be started whenever it is required by some inference. In EPILOG this is achieved by using the metasyntactic devices described in the previous subsection and by providing a single special-purpose function, called "*Apply*," that the QA process knows how to evaluate whenever its arguments are quoted and metavariable-free. It executes a Lisp function of the same name for the given arguments. In particular, to implement the ASK mechanism we added the following axiom to EPILOG's standard knowledge base; this defines knowing that *w*, for a formula *w* containing no free variables, as being true just in case the *knownbyme?* Lisp function returns *yes* for argument *w*:

$$(\forall_{wff}\ w\ ('w\ WithoutFreeVars)\ ((me\ Know\ (that\ w)) \Leftrightarrow \\ ((Apply\ 'knownbyme?\ 'w) = 'yes))$$

In effect, *knownbyme?* implements the ASK mechanism as a recursive QA-process. Note that EL allows for an optional restrictor in quantified statements, and here a restrictor (*'w WithoutFreeVars*) is present. *WithoutFreeVars* is a predicate with one argument denoting an expression, typically specified using quotation, that is true whenever the argument contains no free variables. To evaluate this predicate EPILOG will have another axiom in its knowledge base:

$$(\forall_{wff}\ w\ ('w\ WithoutFreeVars) \Leftrightarrow \\ ((Apply\ 'withoutfreevars?\ 'w) = 'yes))$$

Where *withoutfreevars?* is the Lisp function that detects whether or not an EL expression contains free variables. An alternative to this approach would be to leave the attachment of procedures to EL predicates or functions implicit; however, by using explicit attachment axioms like the above, EPILOG is able to make its own reflective decisions about when to employ particular procedures.

Whenever the QA process encounters a subgoal that contains an equality in which one of the two equated terms is an *Apply*-term, where its arguments are quoted and metavariable-free, EPILOG evaluates it by executing the Lisp function specified as the

first argument of *Apply*, with the arguments provided for it. The result, which must be a quoted EL term, will be substituted for the *Apply* term in the original equality.

Inference in EPILOG

In this section we will describe other characteristics of EPILOG's inference machinery indirectly related to metareasoning and important to EPILOG's overall functioning.

Normalization

Because EL uses nonstandard constructs (e.g., substitutional quantification, quotation, lambda abstraction, and modifiers), the standard FOL normalization to clause form cannot be used. Besides, clause form can be exponentially larger than the original form, for example, for a disjunction of conjunctions. Normalization in EPILOG is based on a term-rewriting system that currently employs a total of fourteen rules. They perform such transformations as moving negations inward, Skolemizing top-level existentials, ordering the arguments of ANDs and ORs, eliminating simple tautologies, and moving quantifiers inward.

As in other reasoning systems, normalization contributes greatly to reasoning efficiency by collapsing classes of "obviously" equivalent formulas into unique (sets of) formulas. As an example, consider the unnormalized form of the statement, "One email in my inbox contains no message":

(∃ *e*0 (*e*0 *AtAbout Now*)
  ((∃ *x* ((*x Email*) *and* (*x In MyInbox*))
    (*No y* (*y Message*) (*x Contains y*))) ** *e*0))

where we have reduced "my inbox" to a constant for simplicity. If this is provided to EPILOG as a fact, then normalization introduces Skolem constants for the existentials *e*0 and *x*, narrows the scope of the episode-characterization operator "**" to exclude atemporal conjuncts, separates implicit conjunctions, and (if we choose to include a rule mandating this) replaces (*No y phi psi*) with (∀ *y phi* (*not psi*)):

(*SK*1 *AtAbout Now*), (*SK*2 *Email*), (*SK*2 *In MyInbox*),
((∀ *z* (*z Message*) (*not* (*SK*2 *Contain z*))) ** *SK*1)

Note that normalization of a goal, unlike normalization of a fact, does not Skolemize existentials, because these serve as matchable variables. In proving a goal with a top-level universal, the universal quantifier may be eliminated and the variable given a unique new name. This step can thus be thought of as the dual of Skolemization.

Inference Graph Handling

EPILOG's QA is in its simplest form a natural deduction back-chaining system. It starts with the initial question and then generates a proof subgoal and a disproof subgoal

(i.e., the negation of the initial question). The QA process maintains an agenda of subgoals based on a binary tree that decides which subgoal to process first. We have not finalized the method of sorting subgoals on the agenda; candidate criteria include: subgoal complexity, their probability, and their level (i.e., the number of inference steps taken to produce a subgoal from the initial question). More testing is required to decide which combination of criteria is effective in the majority of situations.

Each subgoal is first checked to see if it can be simplified:

1. Conjunctions are split into their conjuncts, and each conjunct is handled independently of its siblings until it is solved. When a solution to a conjunct is obtained, it is properly combined with the solutions of the siblings.
2. Disjunctions are split for each disjunct by assuming the negation of the remaining disjuncts. Here we make use of another feature, namely, knowledge base inheritance. Each question is associated with a knowledge base that defines what knowledge can be used to answer the question. When assumptions are made, they are loaded into a new knowledge base (to be discarded at the end of the QA process) that inherits the contents of the knowledge base used before the addition of the assumptions. Currently the consistency of the assumptions is not checked, but problems will be detected from the contradictory answers produced.
3. Implications, $A \Rightarrow B$, are converted into two alternative subgoals: (*not* A), and B assuming A.
4. Equivalences are split into conjunctions.
5. Universally quantified goals are simplified by generating a new constant and unifying the universal variable with it. If the universal quantifier has a restrictor, the restrictor is assumed.

When no more simplifications can be applied, goal-chaining inference is attempted. From each subgoal, one or more keys are extracted and used to retrieve knowledge. These keys are the minimal well-formed formulas embedded entirely by extensional operators such as quantifiers and truth-functional connectives. Each subgoal maintains another agenda that decides which key to use first for retrieval. As in the case of subgoal ordering, we have not yet finalized the sorting criterion. Possibilities are preferring keys that contain more variables, or ones with the least amount of associated knowledge (so as to focus on a quick proof, if one exists).

Naturally, if a retrieved fact exactly matches a goal, then goal-chaining terminates for that goal. In the general case, goal-chaining inferences can be thought of as being resolution-like, except that the literals being resolved may be arbitrarily embedded by extensional operators. As a simple example, suppose that we have a known fact ($\forall$ x (x P) ((x Q) *and* (x R))), that is, every P is a Q and an R. If we use this fact in pursuing a goal of form ((C Q) *and* *phi*) (where C is a constant and *phi* is some well-formed formula), then the derived goal will be ((C P) *and* *phi*). The extensionally

embedded literals that were unified were (*C Q*) in the goal and (*x Q*) in the given fact. If one of these were intensionally embedded, for instance by the reification operator "*that*" (e.g., forming an object of an attitudinal predicate), the unification and hence the goal-chaining inference would not be attempted. (For details, see Schubert & Hwang, 2000.) For each successful inference performed for a subgoal together with a retrieved formula, a child subgoal is attached to this subgoal and the process is repeated (with termination if the derived subgoal is a trivial subgoal: truth or falsity).

The two processes just described (i.e., simplification and inference) construct an inference tree. However, loops and repetitions can occur, worsening performance or preventing success altogether (in case the subgoal selection proceeds depth-first). Therefore, we added two optimizations, the second of which transforms the inference tree into an inference graph:

1. Loop detection: a loop is created when the same subgoal appears twice along an inference branch. In saying that a new subgoal is the "same" as a previous one, we mean that it is expressed by the same EL formula and is associated with the same knowledge base.
2. To avoid doing the same reasoning multiple times, we detect when a new node is generated with a subgoal identical to a previous one on a different branch (thus not forming a loop). If the knowledge base associated with the previous node contains formulas not available at the new node, we treat the new node as unrelated to the previous one. But if the knowledge base associated with the new node is identical with that of the previous one, we connect the two nodes and completely stop further processing of the new node; and if the knowledge base of the new node properly contains that of the previous node, we again connect the two nodes but then continue to process the new node as if the old didn't exist. In case the old node is answered, the answer is propagated to the new node as well, using the inserted connection.

Term Evaluation

Sometimes the answer may contain complex terms, such as functions, instead of a simple result in the form expected by whoever asked the question.

For example, to the question "How old are you?" the system could answer with "The difference in years between 1st of January 1991 and now" instead of actually computing the difference. To deal with this issue, we augment the question with type constraints that define the desired type of the answers. In addition, we employ axioms that describe how to obtain a particular type from another.

For example, the above question in EL becomes:

$$(Wh\ x\ (\exists\ y\ ('x\ RoundsDown\ 'y)\ (\exists\ z\ ('y\ Expresses\ z\ (K\ (Plur\ Year)))\ (\exists\ e\ (e\ \mathbf{At}About\ Now)\ ((z\ AgeOf\ Epi2Me)\ **\ e))))$$

The QA process will then use axioms in its knowledge base that describe how to use the predicate *Expresses* to compute the x that expresses y as a number of rounded down years.

Examples

In this section we describe some of the examples used to test the features of this system. We will point out how metareasoning plays a (major or minor) role in these examples.

Morbini and Schubert (2007) provided a preliminary discussion of the questions "Do pigs have wings?" and "Did the phone ring (during some particular episode E1)?" Previously, such examples could only be handled "Socratically," leading EPILOG through the proofs step by step, whereas now the questions are solved autonomously, as projected in that paper. We will not repeat the details here, but we should reiterate the claim the examples are intended to illustrate: much of our commonsense question-answering, even if not explicitly concerned with metalevel concepts, tacitly relies on metaknowledge about our own cognitive functioning. In the case of the question whether pigs have wings, we claimed that a negative answer depends on the metabelief that our "pig knowledge" is complete with respect to pigs' major body parts (especially very visible ones; for contrast, consider the question "Do pigs have tonsils?"). This autocognitive approach (as we termed it in Morbini and Schubert, 2007) is not only more realistic than the usual default inference approaches, but also more efficient, because it substitutes fast ASK (self-query) checks for potentially unbounded consistency checks. Similarly, the question whether the phone rang, in the case of a negative answer, depends on metabeliefs about how, and under what conditions, we acquire and retain knowledge about audible events in our environment.

One of the most interesting new questions we have tried so far is the question "How old are you?" Though one can easily imagine simple shortcut methods for answering such a question, doing so in a principled, knowledge-based fashion can be nontrivial. Table 17.1 shows a selection of the knowledge used for this question.

The "@ e" construct in the third axiom means "characterizes an episode that is at the same time as e." The reason for introducing axioms for *AtAbout* is that this predicate appears in our interpretation of English present-tense sentences.

Expressed in EL, the question "How old are you?" as previously mentioned becomes:

$$(Wh\ x\ (\exists\ y\ ('x\ RoundsDown\ 'y)\ (\exists\ z\ ('y\ Expresses\ z\ (K\ (Plur\ Year)))\ (\exists\ e\ (e\ \mathbf{AtAbout}\ Now)\ ((z\ AgeOf\ Epi2Me)\ **\ e)))),$$

Table 17.1
Most significant knowledge used to answer the question "How old are you (now)?"

EPILOG's birth date is Jan. 1, 1991:

(($ 'date 1991 1 1) *BirthDateOf Epi2Me*)

If an event happens at the same time as another then it happens at about the same time as the other:

(∀ *x* (∀ *y* (*x SameTime y*) (*x AtAbout y*)))

Meaning postulate relating ** and @:

($\forall_{wff}$ *w* (∀ *e* ((*w* @ *e*) ⇔ (∃ *e*1 (*e*1 *SameTime e*) (*w* ** *e*1)))))

The age of a physical object during an arbitrary episode is the difference between the date of the episode and the date of birth of that object:

(∀ *y* (*y PhysicalObj*) (∀ *x* (*x* (*be* (*BirthDateOf y*)))
(∀ *e* (((*TimeElapsedBetween* (*DateOf e*) *x*) *AgeOf y*) @ *e*))))

Defines how to evaluate the function *TimeElapsedBetween* and to express the result as an amount in a given unit of measure (i.e. *type*, such as "year" or "second"):

(∀ *x* (*x ELDate*)
(∀ *y* (*y ELDate*)
($\forall_{pred}$ *type* ('*type* ELTimePred)
(∀ *r* ('*r* = (*Apply* 'DiffOfDates? '*x* '*y* '*type*))
('*r Expresses* (*TimeElapsedBetween x y*) (*K* (*Plur type*)))))))

Defines how to compute the floor of a given numeric amount:

(∀ *x* (∀ *y* (('*y NumericAmount*) *And* ('*x* = (*Apply* 'RoundsDown? '*y*)))
('*x RoundsDown* '*y*)))

after the addition of implicit conventional constraints on the form of the answer. The answer found is that the age of EPILOG is (*amt* 18 (*K* (*Plur Year*))). This answer is the unifier found for the variable *x* while the unifier found for the variable *z* is (– (*DateOf Now*) ($ '*date* 1991 1 1)). Metareasoning is used to find the expression that represents the same amount of time but is expressed as a number of years. During this process the QA process applies the appropriate conversion and evaluation functions based on the syntactic form of the formulas involved.

This application of metareasoning can be seen as making EPILOG aware of the procedures at its disposal and what they accomplish, leaving to EPILOG's own deliberate decision making the choice of what procedural knowledge to use at what times. This also opens the door to future work on learning by self-programming (see Robertson and Laddaga's work in this vol., chap. 7): the creation and purposeful use of new programs (or plans) aimed at solving specific problems.

The next question considered here is "What is your name (now)?" Metareasoning enters the process only incidentally here (we'll point out where), but to the extent

Table 17.2
Knowledge used to answer the question "What is your name (now)?"

EpilogName is a name:

(*EpilogName Name*)

A name is a thing:
(∀ *x* (*x Name*) (*x Thing*))

Now is during event *E2*:

(*Now During E2*)

The event *E2* is characterized by EPILOG having name *EpilogName*:

((*EPILOG Have EpilogName*) ** *E2*)

Have is a continuous property: if *x* has *y* in *e* then *x* has *y* at all times during *e*:

(∀ *x* (∀ *y* (∀ *e* ((*x Have y*) ** *e*)
(∀ *z* (*z During e*) ((*x Have y*) @ *z*)))))

Meaning postulate relating ** and @:

($\forall_{wff}$ *w* (∀ *e* ((*w* @ *e*) ⇔ (∃ *e*1 (*e*1 *SameTime e*) (*w* ** *e*1)))))

For every time *e* at the same time of *e1* if *e* is during an event *x* then also *e1* is:

(∀ *x* (∀ *e* (*e During x*) (∀ *e*1 (*e*1 *SameTime e*) (*e*1 *During x*))))

that an agent's knowledge makes reference to itself (here, through the self-referring term EPILOG), it indicates its potential for reflective cognition. Table 17.2 lists the knowledge used to answer this question.

The question in EL is represented as

(∃ *e*0 (*e*0 *AtAbout Now*)
((∃ *z* ((*z Name*) *and* (*EPILOG Have z*))
(∃ *y* (*y Thing*) (*y* (*Be* (*L x* (*x* = *z*)))))) ** *e*0))

The apparently convoluted form is due to the fact that this question is automatically generated from the NL input. ("What" is interpreted as "what thing," and the verb phrase "is your name" becomes "is (at about now) identical with a name that you have.") After normalization we obtain the simpler question

(∃ *e*0 (*e*0 *AtAbout Now*)
(∃ *z* ((*z Name*) *and* (*z Thing*)) ((*EPILOG Have z*) ** *e*0)))

The normalization procedure moves inward the "**" operator using the knowledge that "*Name*" and "*Thing*" are atemporal predicates. This knowledge, used by the normalization procedure, is asserted explicitly in EL. This is the incidental use of metaknowledge referred to at the beginning of this example.

Table 17.3
Knowledge used to answer the question "What do you know about the appearance of pigs?"

Pigs are thick-bodied:
((*K* (*Plur Pig*)) *ThickBodied*)
ThickBodied is a predicate about the appearance of something:
(*'ThickBodied AppearancePred*)
Every formula with structure (*x P*) in which *P* is an appearance predicate is a fact about the appearance of *x*:
($\forall_{pred}$ *p* (*'p AppearancePred*) (∀ *x* (*x p*) ((*that* (*x p*)) *AppearanceFactAbout x*)))

In the current reasoning we manually add the fact that EPILOG's name is valid in an interval of time that includes the *Now* point. However, in future we would like this property to be automatically generated by a module in charge of maintaining the self-model of the system.

The last example shows how the metasyntactic devices could be used to answer topical questions. The question is "What do you know about the appearance of pigs?" Table 17.3 contains the knowledge used to answer this question.

The question in EL becomes

(∃ *x* (*x AppearanceFactAbout* (*K* (*Plur Pig*)))) .

The answer found is (*that* ((*K* (*Plur Pig*)) *ThickBodied*)). Note that this answer depends on the metainference from the second and third axioms that the answer well-formed formula is indeed an appearance-fact about pigs. We are not aware of any other system capable of deductive topical reasoning of this sort, in support of descriptive question-answering. However, to retrieve more complex knowledge about pigs, for example, that pigs have curly tails, more complex knowledge would have to be used.

Discussion

The traditional conception of the role of metareasoning in an intelligent agent is diagrammed in figure 1.2 (this vol., chap. 1). The emphasis in this conception is on control and monitoring of object-level reasoning by higher-level reasoning processes, and in turn, the control of action in the world by object-level reasoning (see, e.g., Cox, 2005).

The distinction we have made between procedural knowledge, world knowledge, and metaknowledge might be thought to correspond respectively to the ground-level, object-level, and metalevel modules in figure 1.2. However, this is not so, because as

our examples showed, even question-answering can make simultaneous use of procedures, object-level knowledge, and metaknowledge.

The deployment of these kinds of knowledge is not stratified, but instead tightly intertwined: any subgoal in the QA-process might draw upon procedural, object-level, or metalevel knowledge for its achievement.

It might be thought that this mingling of knowledge and reasoning levels in our system is attributable to our focus on question-answering instead of autonomous, motivated behavior. However, the kind of purposive agent we envisage (and are actively working toward) will still intertwine the various kinds of knowledge and reasoning in the manner of our QA system, rather than using cascaded levels of decision making. It will be based on continual modification, evaluation, and partial execution of a "lifelong" plan, as diagrammed in figure 17.1.

The lifelong plan will contain hierarchically structured goals and actions, and the agent will perpetually try to improve the expected long-term net utility of the plan, while at the same time executing steps that do not require further planning and are currently due. The steps themselves could be reasoning steps (e.g., try to determine the truth or falsity of some proposition) just as easily as they could be communicative or physical actions; so, in terms of the structure and execution of the life plan, there is not a clear separation between thinking and acting. Furthermore, the expansion of goals or high-level steps into executable actions in general will depend on reasoning similar to the reasoning currently carried out by our QA mechanism; that is, this

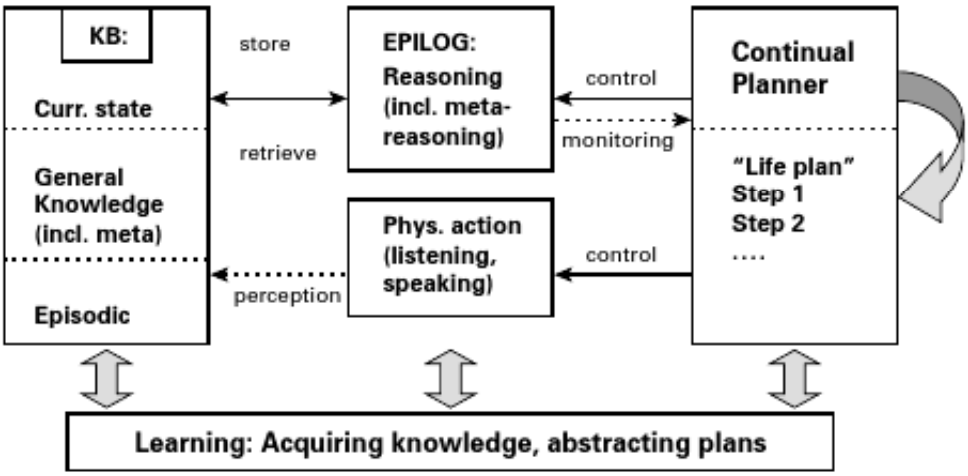

Figure 17.1
Reasoning (including reasoning based on metaknowledge) in a continually planning, self-motivated agent. (No uniform metacontrol.)

reasoning will draw on procedural knowledge, world knowledge, and metaknowledge as needed.

In short, the role played in the standard model of figure 1.2 by metalevel reasoning is played in our conceptual architecture by a continual reward-seeking planner, as in figure 17.1. Though hierarchical planning involves multiple levels of plan structure, the distinction between these levels is orthogonal to that between ground-level, object-level, and metalevel knowledge and activity. Of course, in any behavioral system (as in a human organization), the buck has to stop somewhere; at some root level, there have to be fixed mechanisms that actually make the system run. For us, that mechanism will be a carefully designed continual-planning plus execution algorithm, but exactly what it does will depend on the special-purpose procedures, world knowledge, and metaknowledge that it accesses. Note that this does not mean that all decision making takes place at a single level. The planner may well schedule subplans and procedures that already incorporate decision-making steps. However, the general planner will integrate expected utilities globally, in order to make globally rewarding decisions.

How does this outlook on metareasoning and agent architecture relate to some of the other perspectives found in this book? First, Cox and Raja's manifesto (this vol., chap. 1), along with contributions like those of Costantini, Dell'Acqua, and Pereira (2008), and several others, adhere at least roughly to the schema in figure 1.2, the emphasis being on metalevel control and self-improvement through self-monitoring. As may be inferred from our remarks above on planner-based agency, we suppose that self-improvement is primarily a matter of learning to plan better. These improvements would be a by-product of the processes employed by the fixed planning-and-execution system, including ones for synthesizing, abstracting, and storing subplans, for using available knowledge to predict consequences of contemplated actions and their net future utility, and for using the episodic record of the agent's experiences to debug, improve, and augment the available knowledge about the world, about the agent itself, about the consequences of actions, and about their utilities to the agent.

These activities aimed at self-improvement certainly will involve metalevel reasoning as well as object-level reasoning; but we have not yet explored them in detail, because they seem to us highly dependent on the fundamental knowledge representations and reasoning abilities of the agent, indicating that the latter need to be more fully developed first. For example, planning involves reasoning about the conditions before, during, and after an action is performed, and hence failure analysis is heavily dependent on the expressiveness of the action and state representations used, and on the available reasoning methods. Similarly, the problem of detecting inconsistencies or gaps in an agent's declarative knowledge depends very much on the kinds of representations and reasoning methods employed. This motivates our current focus on the issues of developing representations and reasoning methods that allow for human-

like commonsense knowledge and reasoning, including reasoning about the self and about syntactic objects internal or external to the system.

Unlike Hart and Scassellati (this vol., chap. 18), we are not directly concerned with gaining insight into phenomena associated with self-awareness in humans (or higher mammals), such as self-recognition in a mirror. Our quest is for explicit self-awareness in machines that are knowledgeable and capable of reasoning, though certainly we would like to make use of findings about human self-awareness where possible. The work of Gordon, Hobbs, and Cox (this vol., chap. 19) is closest in spirit to our own, in the emphasis it places on developing a rich, human-like self-model in machines, and on formulating representations adequate for this task, rather than focusing primarily on process. However (and this may just be a terminological difference), they regard metareasoning as concerned chiefly with monitoring and control, whereas we regard the process of drawing conclusions about one's own mental contents and attributes, even in the service of settling such issues as whether pigs have wings, as bona fide instances of metareasoning.

Summary and Future Work

The examples given show the ability of the current system to handle basic forms of metareasoning, in support of commonsense question-answering about the world and about itself. This ability is based on a systematic implementation of an ASK mechanism for self-query, and of a general syntax and mechanism for handling quasi-quotation and substitutional quantification. These enhancements are layered on top of a capacity for inference in a very rich natural logic (EL) that can deal with events characterized by complex sentences, predicate modification, and various forms or reification and modality.

As we showed, the new mechanisms enable well-founded question-answering for such questions as "How old are you?" (using the birth date, and assuring an answer expressed in a syntactically appropriate way), and "What do pigs look like?" (based on syntactic inference about which internal formulas express appearance knowledge about pigs). In addition, the new mechanisms enable deliberate reasoning about special-purpose procedures available to the system and about their goal-directed invocation. We also showed previously how autocognitive reasoning can enable effective negative inferences (e.g., that pigs don't have wings, or that the phone didn't ring in the last fifteen minutes) and inferences about the system's autobiography and recent discourse events (Schubert, 2005; Morbini & Schubert, 2007). Though we have not addressed self-improvement based on metareasoning (the classical goal of metareasoning), we regard the kinds of introspective abilities we have implemented as crucial to such self-improvement. In principle, they enable reasoning about the system's own knowledge and procedures, including their shortcomings.

However, much remains to be done, and the following are some of the more pressing items:

1. So far we have implemented only an exhaustive retrieval mechanism to allow testing of the reasoning system. We will need a much more selective retrieval scheme in order to be able to scale up to a large knowledge base, as needed for commonsense applications. The original version of EPILOG had an efficient content-based indexing scheme, but it was not designed to handle the metasyntactic devices we have added, suffered from certain retrieval gaps (in proving existential goals), and depended on hand-coded type hierarchies rather than forming these automatically. We are working to overcome these limitations, and plan to test the scalability of the resulting system using some large knowledge base (e.g., the FOL version of OpenCyc; see Ramachandran, Reagan, & Goolsbey, 2005).
2. The previous version of EPILOG employed reasoning "specialists" to efficiently perform taxonomic, temporal, partonomic, and other important kinds of specialized reasoning. The specialists were invoked "unconsciously" for predicates and functions they were designed to reason about, but in the new version, we would like to make the inference engine aware of its specialists and their capabilities, using explicit attachment axioms involving the *Apply* function.
3. Another front that needs work is the refinement of the ASK mechanism for knowledge introspection. Currently the mechanism is made time-bounded by a hard limit on depth of reasoning. (However, the limit can be computed, so that several desirable properties of knowing are maintained; e.g., given that EPILOG knows *A* it also knows (*A or B*)).

Acknowledgments

This work was supported by NSF grants IIS-0328849 and IIS-0535105 and by a gift from Robert Bosch Corporation. We also thank the anonymous referees for their helpful comments.

References

Costantini, S., Dell'Acqua, P., & Pereira, L. M. (2008). A multi-layer framework for evolving and learning agents. In M. T. Cox & A. Raja (Eds.), *Metareasoning: Thinking About Thinking, Papers from the AAAI Workshop* (pp. 121–128). Tech. Rep. No. WS-08-07. Menlo Park, CA: AAAI Press.

Cox, M. (2005). Metacognition in computation: A selected research review. *Artificial Intelligence, 169*(2), 104–141.

Kaplan, A. (2000). *A computational model of belief*. Ph.D. dissertation, University of Rochester. Department of Computer Science.

Kaplan, A. N., & Schubert, L. K. (2000). A computational model of belief. *Artificial Intelligence, 120*(1), 119–160.

Morbini, F., & Schubert, L. K. (2007). Towards realistic autocognitive inference. In E. Amir, V. Lifschitz, and R. Miller (Eds.), *Papers from the 2007 AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning* (pp. 114–118). Menlo Park, CA: AAAI Press.

Ramachandran, D., Reagan, P., & Goolsbey, K. (2005). First-orderized ResearchCyc: Expressivity and efficiency in a common-sense ontology. In P. Shvaiko, J. Euzenat, A. Leger, D. L. McGuinness, and H. Wache (Eds.), *Papers from the 2005 AAAI Workshop on Contexts and Ontologies: Theory, Practice, and Applications* (pp. 33–40). Menlo Park, CA: AAAI Press.

Schubert, L. K. (2005). Some knowledge representation and reasoning requirements for self-awareness. In M. Anderson & T. Oates (Eds.), *Proceedings of the AAAI Spring Symposium on Metacognition in Computation* (pp. 106–113). Tech. Rep. No. SS-05–04. Menlo Park, CA: AAAI Press.

Schubert, L., & Hwang, C. (2000). Episodic logic meets Little Red Riding Hood: A comprehensive, natural representation for language understanding. In L. Iwanska & S. Shapiro (Eds.), *Natural language processing and knowledge representation: Language for knowledge and knowledge for language* (pp. 111–174). Menlo Park, CA: AAAI Press.