COMPUTATIONAL MODELS OF HUMAN INTELLIGENCE

CMHI Report Number 2 **16 December 2018**

Self-Aware Problem Solving

Patrick Henry Winston

Computer Science and Artificial Intelligence Laboratory
Center for Brains, Minds, and Machines

I describe a problem-solving scenario in which the Genesis story understanding system tells its own story, in its own inner language, as it answers a question, "Did Lu kill Shan because America is individualistic," about a grisly murder. Genesis's inner-language story enables Genesis to describe, in English, what it is doing as it answers questions, finds concepts in its own thinking, summarizes, instructs, and finds similar problem-solving stories. I suggest that the ideas in Genesis's self-awareness capability will lead to more trustworthy systems.

Keywords: computational models of human intelligence; story understanding; self-aware problem solving.

Self-Aware Problem Solving

Patrick Henry Winston

Computer Science and Artificial Intelligence Laboratory
Center for Brains, Minds, and Machines

16 December 2018

Abstract

I describe a problem-solving scenario in which the Genesis story understanding system tells its own story, in its own inner language, as it answers a question, "Did Lu kill Shan because America is individualistic," about a grisly murder. Genesis's inner-language story enables Genesis to describe, in English, what it is doing as it answers questions, finds concepts in its own thinking, summarizes, instructs, and finds similar problem-solving stories. I suggest that the ideas in Genesis's self-awareness capability will lead to more trustworthy systems.

Keywords: computational models of human intelligence; story understanding; self-aware problem solving

.

Vision: Story telling viewed as the key to self-aware behavior

What does it mean to be self aware? My approach to this question is highly influenced by Marvin Minsky's notion of suitcase words. Words like *intelligence* and terms like *self awareness* are labels for concepts with so many different meanings they are like giant suitcases, so big you can stuff just about anything into them (1988; 2006).

The aspects of consciousness and self awareness I address in this paper have to do with the story I tell myself as I solve problems. Introspectively, I feel like I talk to myself constantly about what is going on as I think about a problem and about what I am doing to solve it. Moreover, I reflect on what I am saying to myself, asking myself questions and working up answers.

Here, I explain how the Genesis story understanding system answers questions about the stories it reads and describes what it is doing in its own inner language. Genesis uses the inner-language description to build an understanding of its own behavior that enables all of Genesis's story-understanding

capabilities to engage, including question answering, concept discovery, summary, instruction, and the retrieval of similar problem-solving stories. In particular, the inner-language description enables Genesis to produce an English monologue at a parameterized level of detail.

Previous reports describe Genesis and cite related work (Winston, 2014; Winston and Holmes, 2018).

Ultimately, like a human problem solver, the Genesis question answerer grounds itself in mechanisms that lie behind what I call an *explanation barrier*. At some level, you can describe how you drink from a cup—move hand, grasp, lift—but then, at some other level, you just contract this muscle or that without being able to describe how. At that level, your problem solving capability grounds itself in motor programs. The Genesis problem solver grounds itself in methods such as those that look for story elements, search for connections, and check beliefs. All these just do what they are supposed to do without explanation; I call them *just-do-it* methods.

Steps: Explanation via story understanding

To enable Genesis to explain itself, I decided to adhere to the following hypothesis:

> **The story-primacy principle:**
> **Question answering is a special case of problem solving,**
> **and problem solving is a special case of recipe following,**
> **and recipe following is a special case of story understanding.**

The story-primacy principle led naturally to the following steps:

- Work out recipe-like problem-solving micro-stories and use them to answer questions.
- Arrange for Genesis to tell itself the story of what it is doing under the guidance of the micro stories.
- Have Genesis use all the inferencing and concept-finding apparatus already in place in support of ordinary story understanding.

As I followed these steps, I knew that I would benefit from prior work on problem solving, especially the work of Pat Langley and his co-authors on what they call the Revised Standard Theory (Langley *et al.*, 2013, 2014; Bai *et al.*, 2015; Langley *et al.*, 2016). The Revised Standard Theory takes to another level the Standard Theory of Allen Newell, John C. Shaw, and Herbert Simon (1958). Langley and his co-authors point out that flexible problem solving requires various parameterizations, such as the depth of subproblem creation allowed and whether the problem solver reacts only to current state, only to the goal state, or to the difference between the current state and the goal. Like the Revised Standard Theory, Genesis makes use of guidance from such parameterizations and adds several that determine how much detail to report as problem solving progresses.

I also knew that I would benefit from prior work on Genesis, especially the work of Hibba Awad in her MEng thesis (2013). She impressively demonstrated what can be done with mental models using a story based on a psychological study by Michael Morris and Kaiping Peng in which they explored cultural differences in attitudes toward violence. (1994). Here is the version read by Genesis:

> Start story titled "Lu murder story/Eastern." I am Asian. Lu is a student. Shan is a student. Lu inhabits America. Lu fails his dissertation defense. Goertz is Lu's advisor. Goertz and Lu are not friends. Lu is Chinese. Lu had highest entrance exam score. Lu is a bachelor. Lu is lonely. Lu owns a gun. Lu practices shooting. Lu passes his second dissertation defense. Lu becomes a lab assistant because Lu does not find a job. Shan graduates with Lu. Shan received national award. Faculty rejected Lu's appeal. Goertz angers Lu. Shan comes from a small Chinese village. Shan is married. Shan is social. Shan has friends. Shan is successful. Shan outperforms Lu. In order to kill Goertz, Lu shoots Goertz. In order to kill associate professor Lu shoots associate professor. In order to kill Shan, Lu shoots Shan. In order to kill Lu, Lu shoots himself.

As shown in figure 1, Genesis understands that that Lu shoots Shan, killing him, but Genesis has no idea why Lu killed Shan. But then, when Genesis is asked, *Did Lu kill Shan because America is individualistic*, Awad's Genesis-based system proceeds as follows:

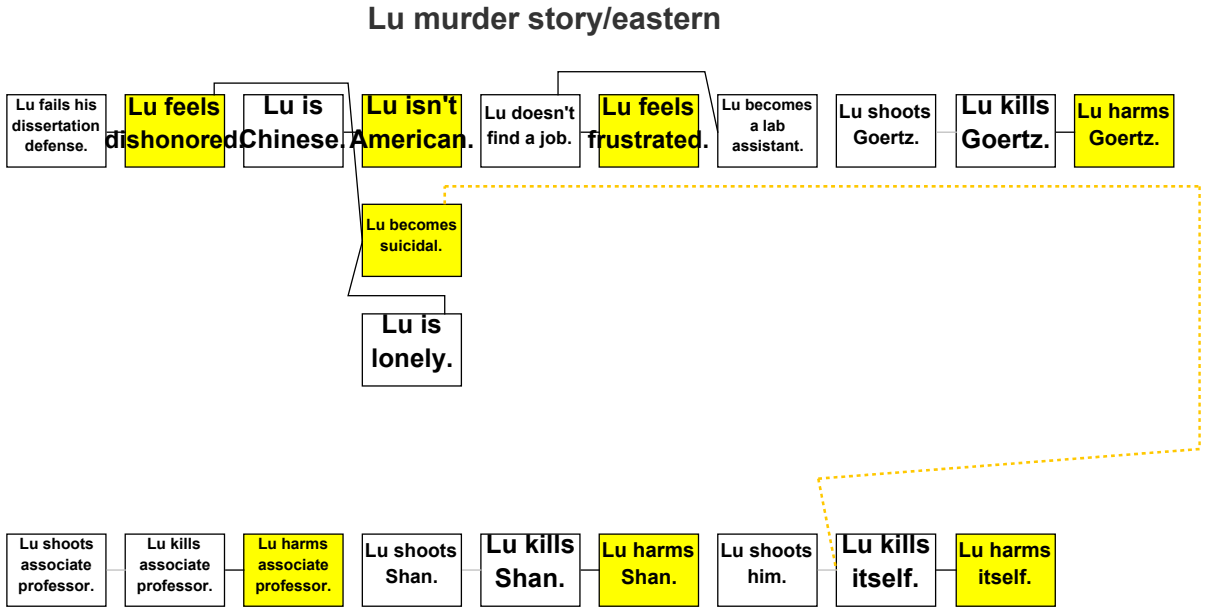

Figure 1: Elaboration graph after reading story Lu murder story. There is no evident reason for why Lu kills Shan.

- Genesis notes that *America is individualistic* is not in the story.
- Genesis, modeling a particular kind of Asian reader, inspects its beliefs, noting that one belief is that America is individualistic.
- Genesis adds *America is individualistic* to the story and notes the consequences.
- Genesis finds a causal path from *America is individualistic* to the murder.
- Genesis answers *yes*.

Note, however, that Awad's Genesis-based system had no ability to explain itself. Enabling that kind of explanation ability is the focus of the work described here.

Genesis uses recipe-like descriptions of what to do

All problem solvers react to problems by doing something. In other problem solvers, the emphasis is on problem spaces and search. In Genesis, the emphasis is on stories, questions, and problems, and how to react to questions and problems with an assortment of short, recipe-like stories that link stories, questions, and problems to a small set of standard story-processing programs.

The short, recipe-like stories fall naturally into several categories. For convenience, I give these categories names, thus augmenting the vocabulary used in previous work on problem solving, especially Langley's. Figure 2 shows how instantiations of the categories fit together.

- *Insights* consist of a problem and an *intention* along with an optional *result*. *Intentions* indicate what is to be done without indicating how.
- *Intentions* connect problems to *approaches* that are appropriate in light of the problem.
- One kind of approach consists of one or more *conditions* and a *method* that is to be performed if all the conditions are satisfied. To see if a condition is satisfied, Genesis tries appropriate checkers. To implement a method, Genesis tries appropriate *executors*.
- *Checkers* determine if conditions are satisfied, and if not, create subproblems aimed at satisfying the unsatisfied conditions. Thus, the Genesis problem solver, like other problem solvers, recurses.
- *Executors* connect methods to just-do-it programs for perception and action.
- *Results* connect solved problems to consequences. A result might, for example, lead to the insertion of a new element into a story, thereby engaging all of Genesis's inferencing and concept discovery apparatus.

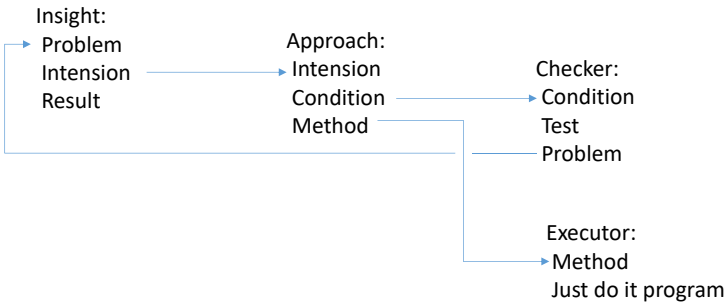

Figure 2: Genesis's problem-solving knowledge consists of short, recipe-like stories.

Insights connect problems to intentions

All the linking knowledge in Genesis is expressed in its own inner language, readily translated from and into English.

In the following, for example, an *insight* links a question, expressed in English as "Did xx cause yy," to what needs to be done, an *intention*, "Establish that a path leads from xx to yy." The line marked with *Success:* identifies an inferred conclusion that follows from successfully answering the question.

```
Insight:

If the question is "Did xx cause yy".

Success: I solved "xx caused yy".

Failure: I did not solve "xx caused yy".

Intention: Establish that a path leads from xx to yy.
```

Note that in this context, an *intention* is an expression of what needs to be done, not a specification for how to get something done. Thus, an intention specifies an end, not a means.

In the Morris-Peng scenario, the question is "Did Lu kill Shan because America is individualistic," which matches "Did xx cause yy," actuating the insight, producing the intention, "Establish that a path leads from America is individualistic to Lu kills Shan."

Approaches connect intentions to methods and conditions

To accomplish what is intended, Genesis uses an *approach*. An approach consists of a method and zero or more conditions that must be established before the method can be applied. For example, to establish a connecting path between two elements, those elements have to be in the story:

```
Approach:

If the intention is "Establish that a path leads from xx to yy".

Condition: Verify that xx is in the story.

Condition: Verify that yy is in the story.

Method: Search for a path from xx to yy.
```

In the Morris-Peng scenario the insight is "Establish that a path leads from America is individualistic to Lu kill Shan" which matches "Establish that a path leads from xx to yy," actuating the approach, producing two conditions, "Verify that America is individualistic is in the story" and "Verify that Lu kills Shan is in the story." If they are satisfied, then the method, "Search for a path from America is individualistic to Lu kill Shan" is tried.

Executors connect methods to programs that lie behind explanation barriers

How does Genesis search for a path? The following *executor* specifies a link to a program.

```
Executor:

If the method is "Search for a path from xx to yy".

Execute: Call "findPath" with xx with yy.
```

The `findPath` program is executable code that produces answers and results for the Genesis problem solver but is not part of the problem solver. Because it is not part of the problem solver, the details of how `findPath` does what it does are inaccessible. By analogy with the idea of abstraction barrier, those details are said to be behind an *explanation barrier*.

The explanation-barrier hypothesis:
All problem solvers eventually ground in programs that just do what they do without explanation about how they do it.

In the Morris-Peng scenario the method is "Search for a path from America is individualistic to Lu kills Shan," which matches "Search for a path from xx to yy," actuating a call to the `findPath` program.

Checkers connect conditions to methods and subproblems

How does Genesis satisfy conditions? Genesis actuates a checker that specifies a predicate method that leads to a just-do-it program, `inStory`, which lies behind an explanation barrier. The `inStory` program just scans through the elements in a story, looking for one element in particular. The line marked with *Success:* identifies a inferred conclusion that follows if the program succeeds.

```
Checker:

If the condition is "Verify that xx is in the story".

Method: Look for xx in the story.

Question: Do I believe xx.


Executor:

If the method is "Look for xx in the story".

Success: I believe xx.

Execute: Call "inStory" with xx.
```

If the test method fails, then the checker recursively tries to solve another problem via an insight that calls for answering the question, "Do I believe xx" by linking that question to an intention that leads to the just-do-it `isBelief` program. Once again, the line marked with *Success:* identifies a inferred conclusion:

```
Insight:

If the question is "Do I believe xx".

Success: I solved "Do I believe xx".

Intention: Establish that I believe xx.

Consequence: Insert xx into the story.


Approach:

If the intention is "Establish that I believe xx".

Method: Look for xx in my beliefs.
```

```
Executor:

If the method is "Look for xx in my beliefs".

Execute: Call "isBelief" with xx.
```

In the Morris-Peng scenario, one condition is "Verify that America is individualistic is in the story," which matches the checker, "Verify that xx is in the story," ultimately actuating a call to the `inStory` program, which fails. But having failed the test, the question "Do I believe America is Individualistic" is posed. The question then actuates an approach because the intention "Establish that I believe America is individualistic" matches "Establish that I believe xx". This ultimately actuates a call to the `isBelief` program, which succeeds, satisfying the condition.

Another condition is "Lu kills Shan," which also matches the checker "Verify that xx is in the story". This time, however, the actuation of the `inStory` program succeeds, satisfying the condition.

Results connect consequences to methods

Note that the approach triggered by the `"Establish that I believe xx"` intention includes a `Consequence:` specification. The consequence actuates a method that leads to an executor that calls a just-do-it program:

```
Result:

If the consequence is "Insert xx into the story".

Method: Insert xx into the story.


Executor:

If the method is "Insert xx into the story".

Execute: Call "insert" with xx.
```

In the Morris-Peng scenario, the result, "Insert America is individualistic into the story," ultimately leads to a call to the `insert` just-do-it program. The insertion triggers ordinary story understanding inference rules that collectively create a path from "America is individualistic" to "Lu kill Shan" as shown in the complete elaboration graph shown figure 3, and with less clutter in figure 4 where only the relevant parts of the elaboration graph are shown.

With both conditions satisfied, and the complete path constructed, the `findPath` just-do-it program succeeds, so the corresponding method succeeds, so what is intended succeeds, so the answer to the question is *yes*.

Genesis tells itself stories as it answers questions

The elaboration graph in figure 5 describes what is going on in Genesis's model of itself as it deals with the Morris-Peng scenario.

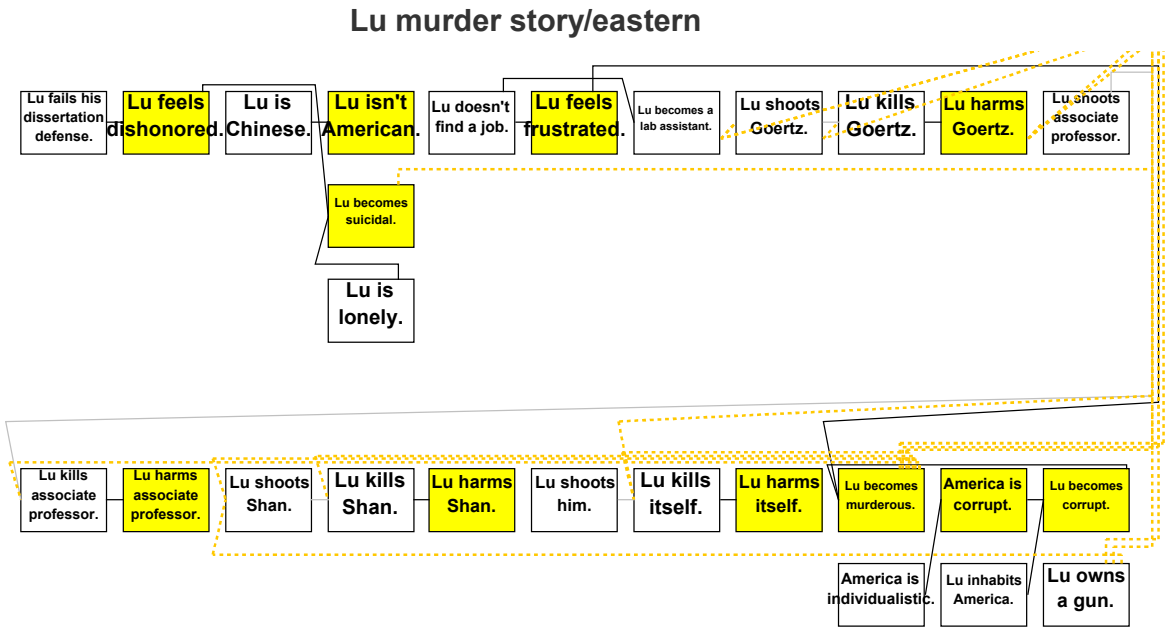

Figure 3: Elaboration graph after asking *Did Lu kill Shan because America is individualistic*.

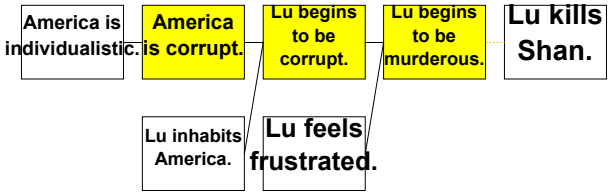

Figure 4: Trimmed elaboration graph after asking *Did Lu kill Shan because America is individualistic,* emphasizing the connecting path, America is individualistic →America is corrupt → Lu becomes corrupt → Lu becomes murderous → Lu kills Shan.

Genesis makes inferences as it tells itself stories

In figure 5, you see that Genesis has made several inferences, all shown in yellow. Problem solving successes produce two by-product inferences: because Genesis used an executor to find "Lu kills Shan" in the story, Genesis believes Lu kills Shan; because Genesis used insights to answer questions, Genesis twice notes that it solved a problem.

Also, Genesis uses standard story-understanding inference rules to make two more inferences: Because Genesis believes "Lu kills Shan," Genesis concludes that Genesis thinks about Lu's killing Shan; because Genesis thinks about something, Genesis concludes, following Descartes, that Genesis exists. The inference rules that do this work, helping to develop the inner story, are as follows:

```
If xx believes ll then xx thinks about ll.

If xx thinks about ll, then xx be.
```

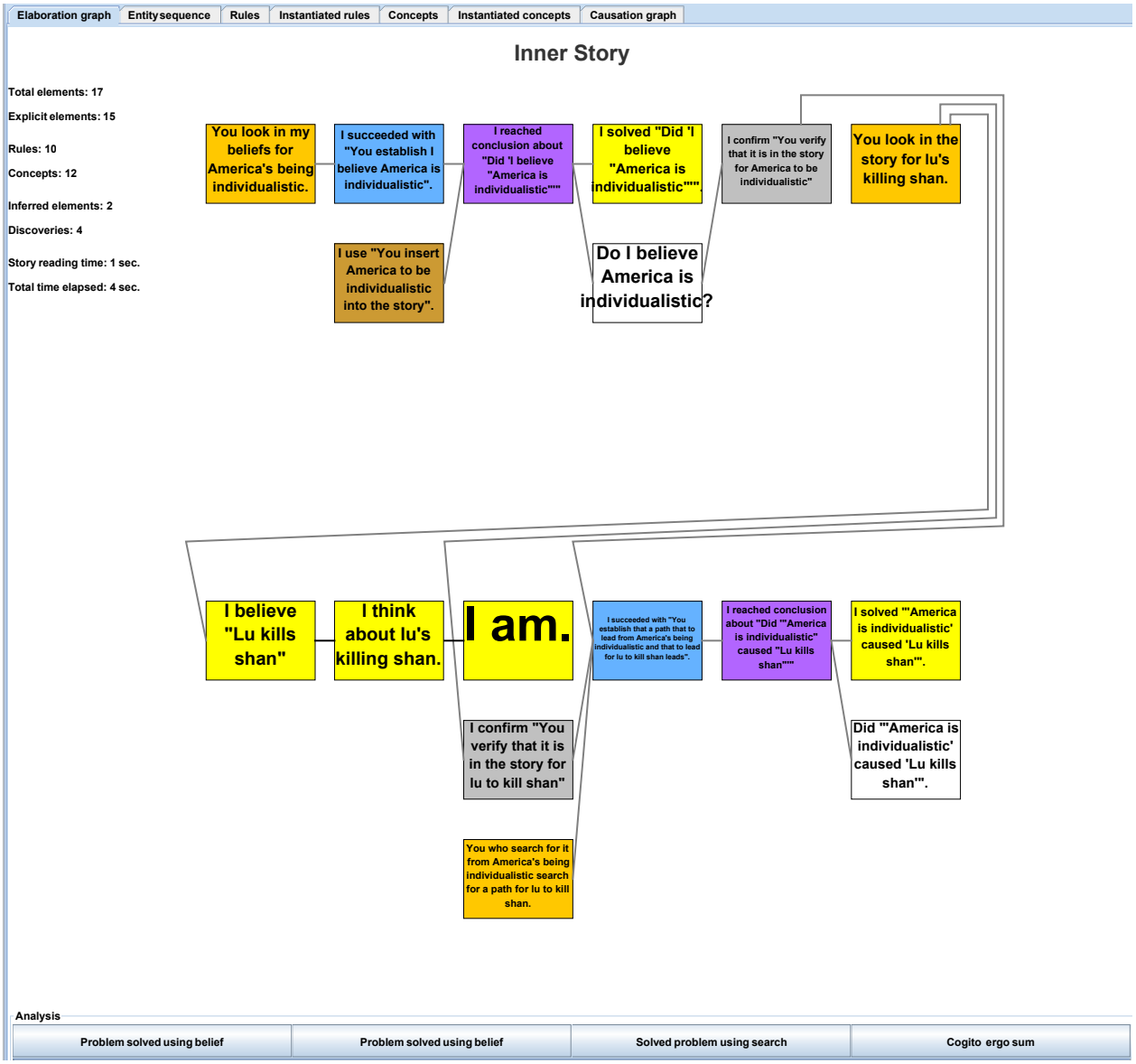

Figure 5: The elaboration graph of Genesis's self story, recounting a problem-solving experience. Color key: white = question; purple = question answered; blue = intention fulfilled; gray = condition satisfied; orange = method succeeded; brown = result noted; yellow = inference made.

Genesis finds concepts in its own self-told stories

Once the inner story is developed, then Genesis uses its standard concept pattern mechanism to find concepts. Figure 6 shows *Problem solved using belief*. The *Problem solved using belief* concept description is:

```
Start description of "Problem solved using belief".

My succeeding with "You establish that i believe vv" leads to my solving rr.

The end.
```

Figure 7 shows *Cogito ergo sum*. The *Cogito ergo sum* concept description is:

```
Start description of "Cogito, ergo sum".

My thinking about vv leads to my being.

The end.
```

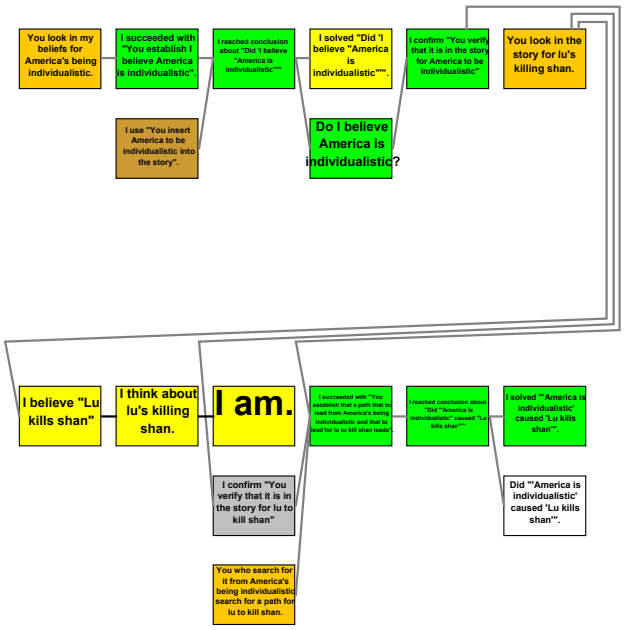

Figure 6: The elements highlighted in green are all part of a "Problem solved using belief" concept derived from the *I succeeded with "... I believe ... → I solved ... caused ...* connection.

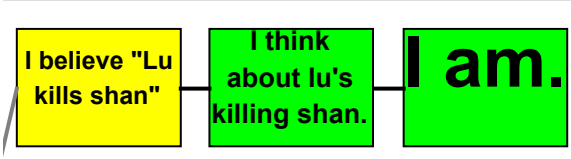

Figure 7: The elements highlighted in green are all part of a "Cogito ergo sum" concept derived from the *I think about... → I am* connection, which is a joke, sort of.

Thus, Genesis reflects on its own inner story just as it reflects on stories in general. Because Genesis tells its inner story in the same inner language it uses for story understanding in general, Genesis can deploy its full armamentarium of story understanding capability on those problem-solving inner stories. It can not only make inferences and find concepts in its own thinking, it can also summarize the self-story, explain with a listener model, and find conceptually similar solutions in memory.

Genesis self-aware story telling enables Genesis to explain what it is doing

Because Genesis tells itself a story in its own inner language, Genesis can compose an inner monologue describing what it is doing, as it does it, based on that inner-language story. Consequently, Genesis

can report on any desired aspect of what it is doing by translating that inner monologue into English.

For example, if Genesis is parameterized so as to have the processing of each insight report on what it is doing and what happens, Genesis handles the Morris-Peng scenario as follows (the abundant quotation marks are characteristic of Genesis's START-based generator (Katz *et al.*, 2005)).

```
I think about "Did '"America is individualistic" cause "Lu kills Shan"'".

     I think about "Did 'I believe "America is individualistic"'".

    I conclude "I believe 'America is individualistic'".

I conclude "'America is individualistic' cause 'Lu kills Shan'".
```

Parameterized another way, Genesis reports only on the top-level problems solved:

```
I conclude "'America is individualistic' cause 'Lu kills Shan'".
```

Parameterized yet another way, Genesis reports on not only what the insights do and what is decided, but also on checker confirmations:

```
I think about "Did '"America is individualistic" cause "Lu kills Shan"'".

     I think about "Did 'I believe "America is individualistic"'".

    I conclude "I believe 'America is individualistic'".

   I confirm "You verify it is in the story for America to be individualistic".

   I confirm "You Verify it is in the story for Lu to kill Shan".

I conclude "'America is individualistic' cause 'Lu kills Shan'".
```

Genesis answers myriad questions, grounding itself in just-do-it methods

Genesis answers many questions, always explains how it answers, not just *Did Lu kill Shan because America is individualistic*:

Question: Was Lu lonely?
Answer: I conclude "Lu is lonely".

Question: How did Lu kill Shan?
Answer: In order to kill Shan, Lu shoots Shan.

Question: Why did Lu kill himself?
Answer: Lu kills itself because Lu is lonely, Lu fails his second dissertation defense, Lu inhabits America, Lu fails to find a job, and America is individualistic.

Question: Did Lu kill Shan after Lu killed the Associate Professor?
Answer: I conclude "Lu kills Shan after killing associate professor".

Question: Did Lu kill Shan before Lu killed the Associate Professor?
Answer: I conclude "Lu kills Shan after killing associate professor".

Question: What happened after Lu fails his dissertation defense?
Answer: Lu kills himself because Lu fails the his second dissertation defense.

Question: What happened after Lu became corrupt?
Answer: Lu kills goertz, Lu kills associate professor, Lu kills Shan, and Lu kills himself because Lu becomes corrupt.

Question: What happened before Lu became corrupt?
Answer: Lu becomes corrupt because Lu inhabits America, and America is individualistic.

Question: What happened when Lu shot himself?
Answer: Lu kills himself because Lu shoots himself.

Question: What is the story about?
Answer: Story is about murderous influence.

Question: What is unexplained?
Answer: Lu inhabits America, Lu fails, Lu is Chinese, Lu is lonely, Goertz angers Lu, and America is individualistic.

The most complicated effort follows from the question, "Did Lu kill Shan before Lu killed the Associate Professor?" Genesis first tries to work with an intention, "I check whether Lu killed the Associate professor is after Lu killed Shan." Failing to find a useful approach, Genesis tries another question, "Did Lu Kill Shan cause Lu killed the Associate professor?" Failing to find a useful insight, Genesis finally asks "Did Lu kill the Associate Professor before Lu kill Shan?" which leads to a contradiction, Here is the governing insight:

```
Insight:

If the question is "Did xx before yy".

Intention: Establish whether yy is after xx.

Question: Did xx cause yy.

Contradiction: Did yy before xx.

The end.
```

The resulting monologue, in full detail, contains 42 remarks, some, alas, in awkward English. Using a parameter to restrict the monologue to questions asked and conclusions reached leads to the following:

```
I ask "Did 'Lu kills Shan before killing associate professor'".

 I don't reach conclusion about Lu kills Shan before killing associate professor.

  I ask "Did '"Lu kills Shan" cause "Lu kills associate professor"'".

  I did not conclude "'Lu kills Shan' cause 'Lu kills associate professor'".

  I ask "Did 'Lu kills associate professor before killing Shan'".
```

```
  I conclude "Lu kills associate professor before killing Shan".

I did not conclude "Lu kills Shan before killing associate professor".
```

In summary, Genesis answers questions and explains what it is doing as story-centered recipe-following problem-solving exercises, grounded in collections of just-do-it predicates such as `inStory`, `isBelief`, and `isInOrder` along with just-do-it primitive methods such as `findPath`, `findRecipe`, `findCauses`, `findConsequences`, and `insert`.

Genesis's self-aware behavior offers a step toward greater trust

Recently, much has been written in the media about the dangers of super-smart programs. I think the best way to address the public's concerns is to ensure that the programs of the future are equipped with an ability to explain what they are doing and how they work. Neural nets, deep or otherwise, cannot. Neither can Watson or Deep Blue.

By contrast, Genesis has started to explain what it is doing and how it works. I expect it will lead to systems that explain themselves on many levels. At some level, they will all have to answer "I don't know how I did it, I just did it." But inasmuch as much of thinking is in our story competence, the intelligent programs of the future, the ones enabled by the ideas in Genesis, will be the ones people will be more inclined to trust.

Contributions

- Noted that *consciousness* and *self awareness* are what Minsky labeled suitcase words.
- Claimed that a problem solver is self-aware, noting that *self aware* is a suitcase term, if it can tell and understand its own story.
- Demonstrated that the Genesis story telling, understanding, and composition system exhibits self-aware problem-solving behavior via a question-answering example involving belief inspection, story augmentation, and search.
- Introduced the insight, approach, checker, executor, and consequence vocabulary for recipe-like stories, along with the intention, method, condition, and consequence linking elements.
- Suggested that self-aware systems offer a step toward trustworthy systems.

Acknowledgements

The work described in this paper was supported, in part, by the Air Force Office of Scientific Research, contract number FA9550-17-1-0081. Other sponsors of work on aspects of the Genesis Group's work include the National Science Foundation, via the Center for Brains, Minds, and Machines, STC award CCF-1231216.

References

Hiba Awad. Culturally based story understanding. Master's thesis, Electrical Engineering and Computer Science Department, MIT, Cambridge, MA, 2013.

Yu Bai, Chris Pearce, Pat Langley, Mike Barley, and Charlotte Worsfold. An architecture for flexibly interleaving planning and execution. *Advances in Cognitive Systems*, 2015.

Boris Katz, Gary Borchardt, and Sue Felshin. Syntactic and semantic decomposition strategies for question answering from multiple resources. In *Proceedings of the AAAI 2005 Workshop on Inference for Textual Question Answering*, 2005.

Pat Langley, Miranda Emery, Michael Barley, and Christopher J. MacLellan. An architecture for flexible problem solving. *Advances in Cognitive Systems*, 2013.

Pat Langley, Chris Pearce, Mike Barley, and Miranda Emery. Bounded rationality in problem solving: Guiding search with domain-independent heuristics. *Mind and Society*, 13, 2014.

Pat Langley, Chris Pearce, Yu Bai, Charlotte Worsfold, and Mike Barley. Variations on a theory of problem solving. *Advances in Cognitive Systems*, 2016.

Marvin Minsky. *The Society of Mind*. Simon and Schuster, New York, NY, 1988.

Marvin Minsky. *The Emotion Machine*. Simon and Schuster, New York, NY, 2006.

Michael W. Morris and Kaiping Peng. Culture and cause: American and Chinese attributions for social and physical events. *Journal of Personality and Social Psychology*, 67(6):949–971, 1994.

Allan Newell, J. C. Shaw, and Herbert A. Simon. Elements of a theory of human problem solving. *Psychological Review*, pages 151–166, 1958.

Patrick Henry Winston and Dylan Holmes. The Genesis Enterprise: Taking artificial intelligence to another level via a computational account of human story understanding. MIT DSpace, Computational Models of Human Intelligence Community, CMHI Report Number 1, 2018. URL http://hdl.handle.net/1721.1/119651.

Patrick Henry Winston. The genesis story understanding and story telling system: A 21st century step toward artificial intelligence. Memo 019, Center for Brains Minds and Machines, MIT, 2014.