20 CHAPTER 1

In The Vengeful Princess, MINSTREL creates a story scene to introduce the reader to the main character of the story:

Once upon a time there was a Lady of the Court named Jennifer...

and story scenes to resolve character fates:

Jennifer was buried in the woods. Grunfeld became a hermit.

These scenes are created to improve the story presentation. The introductory scene is created to ease the reader's transition into the story by providing an immediate identification of the main character and the story genre. The final scenes are created to resolve the reader's expectations about the fates of the story characters and create a sense of closure.

MINSTREL achieves a number of difficult tasks in presentation. These tasks include:

• How introduction scenes are created.
• How denouement scenes are created.
• How story events are ordered, both implicitly and explicitly.
• How paragraphing is used to reflect thematic structure.
• How natural language is generated from concepts.

Further information about how MINSTREL achieves its presentation tasks can be found in (Turner, 1993).

1.4 A Reader's Guide

The remainder of this volume is divided into three sections.

Chapter 2 describes MINSTREL's model of creativity, how creativity is used to augment case-based reasoning, and gives examples of MINSTREL's creativity in planning and storytelling. Chapter 3 also discusses a variety of related issues, including errors in creativity and learning.

Chapter 3 introduces MINSTREL's model of storytelling. Chapters 4, 5, and 6 describe MINSTREL's author-level goals and plans for theme, drama, and consistency.

Chapter 7 presents an extended example of how MINSTREL tells a story. A detailed trace of MINSTREL creating a story is analyzed to reveal the exact sequence of author goals and plans that leads to a finished story.

Finally, Chapter 8 evaluates the MINSTREL model by comparing it to previous work in psychology and artificial intelligence and by analyzing MINSTREL's performance as a storyteller and creator. Chapter 9 contains conclusions and some thoughts on future work.

2.1 Introduction

Creativity—the bringing forth of an original product of the human mind—is the pinnacle of human cognition. Artists, writers, and scientists are treated with intellectual awe, and the word itself brings to mind the roll of men whose discoveries changed their social and scientific cultures: William Shakespeare, Albert Einstein, Leonardo da Vinci, Thomas Edison.

And yet creativity has a mundane side as well. From small child to adult, we are all creative in every aspect of our lives. Faced with a problem we have never before encountered, we combine past knowledge in new ways to create a solution. We fix our cars using hangers and baling wire, invent jokes based on the latest domestic crisis, and make up bedtime stories for our sons and daughters. Far from being the sole province of extraordinary thinkers, the ability to create new solutions to problems is one of the cornerstones of human problem solving.

To understand human cognition, it is essential that we understand the processes of creativity: the goals that drive people to create and the mechanisms they use to create. And because creativity spans the intellectual spectrum from the highest peaks to everyday cognition, it is likely that studying creativity will provide special insights into human thought.

Understanding creativity also has practical value to computer scientists. Current computer programs have little adaptability. Faced with new situations, they act incorrectly or fail completely. Implementing the fundamental processes of creativity in computer programs has the potential to greatly increase the scope and power of the modern computer.
This chapter presents a process model of creativity. A case-based problem solver is augmented with evaluations that act as a creative drive, and a set of creativity heuristics called Transform-Recall-Adapt Methods (TRAMs). TRAMs create new solutions to problems by transforming problems into slightly different problems, solving the new problems, and then adapting any solutions found to the original problem. Repeated application of TRAMs enables the problem solver to make elaborate problem transformations by small steps, invent new solutions substantially different from old ones, and greatly extend the range of problems that can be solved.

The second part of this chapter discusses the implementation of this model in MINSTREL. Although MINSTREL is primarily a storytelling program, MINSTREL’s model of creativity is independent of the storytelling domain. Consequently, the creative problem solving portion of MINSTREL can be applied to other domains and problem tasks. This chapter presents examples of MINSTREL’s model of creativity applied to different problems, and discusses some of the important issues of modeling creativity.

2.2 Creativity and Problem Solving

The goal of this research is to develop a model of creativity and answer the question: What are the processes of creativity common to many problem solving domains? The first step to answering this question is to understand what it means to call the solution to a problem “creative.”

According to psychologists, people recognize a solution as creative if it (1) has significant novelty and (2) is useful (Koestler, 1964; Wallas, 1926; Weisberg, 1986).

We all recognize that creative solutions must be original. They must be new and different from old solutions. But the differences must also be significant. If an artist were to paint the Mona Lisa in a red dress instead of a blue one, the resulting painting would not be considered creative, despite its differences from the original. Significant novelty distinguishes creative solutions from ones that are only adaptations of old solutions.

Usefulness is the second characteristic of a creative solution. We expect problem solvers to be capable: They must develop solutions that solve their problems. Replacing a flat tire with an air raft is novel but not creative, because it doesn’t effectively solve the original problem. Creativity must be purposeful and directed. Creative solutions must have bearing and utility on the problems to which they are applied.

Creative solutions are, therefore, both useful and significantly novel. How are these features reflected in the creative process? Consider the following anecdote concerning the 7-year-old niece of the author:

One day, while visiting her grandparents, Janelle was seated alone at the dining room table, drinking milk and eating cookies. Reaching for the cookies, she accidently spilled her milk on the table. Since Jannel had been recently reprimanded for making a mess, she decided to clean up the spill herself.

Janelle went into the kitchen, but there were no towels or paper towels available. She stood for a moment in the center of the kitchen thinking, and then she went out the back door.

She returned a few minutes later carrying a kitten. The neighbor’s cat had given birth to a litter about a month ago, and Janelle had been over to play with the kittens the previous day. Janelle brought the kitten into the dining room, where he happily lapped up the spilled milk.

Most people find Janelle’s solution to her problem creative. The use of a kitten as an agent and the substitution of “consumption of milk” for “removal of milk” are significant differences from Janelle’s known solution to the “spilled milk problem”, and the success of the new solution shows its usefulness. Janelle’s example illustrates three important principles of creativity:

Creativity is driven by the failure of problem solving.

When Janelle could not find a towel, her old plans for cleaning up a spill became unusable, forcing her to invent a new solution. The need for creativity arises from the failure of problem solving (Weisberg, 1986).

Creativity is an extension of problem solving.

Janelle’s solution to the spilled milk problem arises as part of her problem solving process, and even the creative aspects of her solution—using a kitten as an agent and substituting consumption of the milk for removal of the milk—are broken down into subtasks in the manner of a classical problem-solving strategy (divide and conquer). Although some theorists (e.g., Koestler, 1964; Wallas, 1926) have suggested that creativity is a process fundamentally different from problem solving, there is ample evidence to indicate that creativity is an outgrowth or extension of problem solving (Weisberg, 1986).

New solutions are created by using old knowledge in new ways.

Creative solutions do not spring forth newborn from the head of Zeus; they make use of what the problem solver already knows. Janelle knew that the neighbors had kittens, that kittens like milk, and that the goal “consumption of milk” could subsume the goal of “removal of milk,” and she combined this information into a new solution by using her general knowledge about agents. The significant novelty of creative solutions arises from the problem solver’s
application of knowledge in a new way to the problem (Koestler, 1964; Weisberg, 1986).

The model of creativity presented in this chapter incorporates these principles into a model of case-based problem solving.

2.3 Case-Based Models of Problem Solving

MINSTREL’s model of creativity is built on a case-based model of problem solving (Hammond, 1988; Kolodner, 1987; Schank, 1987).

Case-based reasoning systems are driven by an episodic memory of past cases rather than a base of inference rules or knowledge-intensive heuristics. Given a problem, a case-based problem solver (1) recalls a past problem with the same features and its associated solution, (2) adapts the past solution to the current problem, and then (3) assesses the result. To drive home from work, a case-based problem solver recalls a previous time he or she drove home and the route he used, modifies the route in light of changing road conditions (perhaps a particular street is closed for repair) and then assesses that route according to his or her general knowledge about driving.

This model is illustrated in Figure 2.1. Problem specifications are input at the left side, where they are used to recall similar past problems. The solutions from these past problems are then adapted to the current problem. Finally, the adapted solutions are assessed to determine if they are useful and meet other domain-specific considerations (e.g., a mechanical engineer might want to create efficient solutions). (A more complete review of case-based reasoning can be found in Slade, 1991.)

Case-based problem solving has several benefits. First, case-based problem solving is similar to the reasoning done by human experts (Riesbeck & Schank, 1989). Expertise depends on experience, and case-based reasoning captures this connection.

Second, case-based problem solving explains how a problem solver can use new experiences to extend his knowledge. Planning and storytelling systems that capture problem solving knowledge as static rules (i.e., Lebowitz, 1985; Meehan, 1976; Warren, 1978) require additional mechanisms to explain how that knowledge can be extended.

A final advantage of case-based problem solving is efficiency at solving routine problems. By recalling a past problem that shares all the features of the current problem, the problem solver is assured (1) that the solution from the recalled problem will apply to the current problem, (2) that adapting the recalled solution to the current problem will be simple, (3) that little or no assessment of the adapted solution will be necessary, and (4) that the recalled solution is very likely to work. Re-using past solutions under identical circumstances requires little effort and results in a high degree of success.

However, case-based problem solving has an obvious shortcoming: It fails when faced with a new problem. If a similar past problem cannot be recalled, the case-based problem solver has no way to discover, build, or create a new solution. Using old solutions over and over again is acceptable and efficient in many domains. It would be tiresome, for instance, to create a new route home from work each day. But when no old solution is known, a problem solver must be able to create a new solution. Furthermore, case-based problem solving has difficulty explaining domains such as art and literature. In these domains, problem solvers often reject solutions simply because they are old or repetitive.

To capture the creative process, case-based problem solving must be extended to include (1) a creative drive, and (2) processes that can combine old knowledge in new ways to create problem solutions.

2.4 Failure-Driven Creativity

People are conservative problem solvers. They expend the effort to create new solutions when familiar, known solutions fail (Weisberg, 1986). Janelle created a new solution to the spilled milk problem only because the lack of a towel prevented her from applying an old plan.

Old solutions may fail for a number of reasons. They may not solve the problem, or may solve the problem inefficiently. Or they may be rejected simply because they are old, as in literature and art, where a premium is placed on new ideas. Whatever the cause, it is the failure of old solutions that drives the problem solver to create a new solution. A model of creative problem solving needs a mechanism for detecting planning failures and using that to drive the creative process.

Case-based problem solving can fail at each of the three steps shown in Figure 2.1: (1) if a past problem situation similar to the current problem situation cannot be recalled, (2) if a recalled solution cannot be adapted to the current problem, or (3) if the adapted solution fails a domain assessment. In the
MINSTREL model of creativity, a failure at any of these steps causes MINSTREL to attempt to create a new solution to the problem. Like Janelle, MINSTREL is creative when it encounters a problem for which its past solutions fail.

Because MINSTREL is a model of storytelling as well as creativity, MINSTREL also implements an artistic drive to create. Unlike their counterparts in traditional problem solving domains such as engineering, artists sometimes reject solutions even when problem solving is successful. Artists create for the sake of creation; they find repetition of solutions boring and unacceptable. (In fact, all human problem solvers get bored with repetitive solutions, but particular emphasis is placed on this motivation in the arts.) MINSTREL implements the artistic drive to create as a boredom assessment.

The boredom assessment operates during the Assess step of problem solving (see Figure 2.1). The boredom assessment examines proposed problem solutions to determine if they've been used too many times previously, that is, have become boring. To do this, MINSTREL uses episodic memory.

Episodic memory is the autobiographical record of the events and experiences that make up a person's personal history (Cohen, 1989; Tulving, 1972). MINSTREL uses a model of episodic memory based on work by Schank (1982) and Kolodner (1984), and elaborated and tested by Reiser, Black, and Abelson (Reiser, 1983, 1986; Reiser, Black, & Abelson, 1985), who term it the "context plus index" model. In this model, episodes are organized according to their distinctive differences. Two episodes with significant differences fall into different memory categories and will not be recalled together. Episodes with no significant differences fall into the same memory category and are recalled as a group.

To determine if a solution has become boring, MINSTREL indexes the solution in episodic memory and counts how many times similar solutions have been used. If the solution has been used more than a small number of times, it is judged boring and rejected.

To illustrate this process, suppose that MINSTREL is building a story scene in which a knight's life is endangered. MINSTREL's episodic memory contains scenes about King Arthur and his Knights. Some of these scenes are initially seeded into MINSTREL's memory, as if MINSTREL had read several short stories about King Arthur. Others are from stories MINSTREL invented during earlier storytelling sessions. In this example, MINSTREL's episodic memory contains only one scene. In this scene, a knight fights a dragon. To build a new scene in which a knight's life is endangered, MINSTREL tries to recall a similar scene from a previous story. This recalls the scene in which a knight fights a dragon. Because the recalled scene had only been used once before, it is judged "not boring" and used as the basis for the new scene. Then the new scene is added to the current story and to episodic memory.

Now episodic memory contains two scenes in which a knight fights a dragon. The next time MINSTREL creates a scene in which a knight fights a dragon (perhaps again to solve the "build a scene in which a knight is endangered" problem), the boredom assessment will reject the scene. "Knights fighting dragons" has become boring, and MINSTREL will be driven to create a new way in which a knight can be endangered, even though problem solving succeeded in finding a useful solution to the original problem.

As MINSTREL tells stories and indexes them in memory, MINSTREL's storytelling behavior changes. MINSTREL may tell stories about knights fighting dragons for a while, but these soon become boring and MINSTREL moves on to other topics. Thus, the boredom assessment models the artistic drive to create. MINSTREL, like any human faced with a repetitive task, becomes bored with the "known" solutions, and is driven to create new solutions. And because of its creativity heuristics, MINSTREL can invent new solutions when it becomes bored with old ones.

MINSTREL's boredom heuristic is simple. It ignores the type of problem being solved and other constraints that might realistically affect an artist's determination of whether a concept or solution should be reused. A storyteller, for example, will not want to create new solutions for every character action in a story. How a knight gets from place to place is probably unimportant, and any known solution will do, no matter how frequently it has been used in the past. And in other domains there will be similar considerations.

For storytelling, MINSTREL applies the boredom heuristic only when creating the story events that illustrate the plot of the story. The plot, which is the sequence of story events that illustrate the theme or most important point of the story, is the most important element of the story, and so MINSTREL strives to be creative when building the plot. Other elements of the story, such as events that were added to keep the story causally consistent, are of less importance, and so MINSTREL accepts noncreative solutions for these problems.

For any particular artistic domain, then, there will be other considerations in determining whether a concept is "boring." But the basic consideration will remain the novelty of the concept: how often that concept or a similar one has been used in the past. The basic process behind MINSTREL's boredom heuristic—using episodic memory to determine the novelty of an idea—supports this basic need and has the flexibility to be augmented into a more complete and realistic boredom assessment.

### 2.5 The Challenges of Creativity

By what processes are new solutions created? As Janelle's example illustrated, people invent new solutions by (1) recalling knowledge not part of the known solutions to the problem, and (2) adapting that knowledge to create a solution (Weisberg, 1986). Four factors make this a difficult task.

First, the obvious source of knowledge about the current problem—similar
past problems—has been exhausted. Case-based problem solving tries to find solutions indexed under similar past problems. But the creative process begins when problem solving fails; the solutions indexed under the current problem have already been tried and rejected. To discover a new solution, the problem solver must find knowledge not indexed under the current problem. The difficulty is knowing where in the space of episodic memories to seek knowledge. Knowledge grabbed willy-nilly from memory will be unlikely to apply to the current problem.

New solutions require knowledge not indexed under the current problem.

Second, the problem solver may not even have any incorrect or partial solutions on which to base his problem solving efforts. If the current problem is sufficiently different from past problems, then case-based problem solving may have failed to recall any similar past problems and their associated solutions. Thus it is difficult for the creative problem solver to search for new knowledge in the solution space, because he may not have even an incorrect solution from which to start. However, he will always have his problem description, so he can search the problem space. In general, the problem solver must find new knowledge by searching the problem space. The problem solver must change the current problem into some new problem, in hopes of finding useful knowledge in the recalled solutions to the new problem. This casts the problem of creativity in a new light: The first step to creativity is changing the problem, not the solution.

Creativity involves recasting the problem.

Third, the complexity of adapting the discovered knowledge may overwhelm the creative effort. If Janelle had attempted to adapt her plan for the "getting to school" problem to the "spilled milk" problem, the difficulty of the adaptation task would be insurmountable. (How can knowledge about riding a school bus be applied to the problem of cleaning up spilled milk?) Trying to solve a difficult problem by substituting an impossible problem is a poor strategy. Somehow the creative process must limit the complexity of the adaptation process.

Adaptation must avoid too much complexity.

Finally, the creative process must be capable of discovering solutions substantially different from the original solution. There is a need to limit the creative process to simplify the search for new knowledge and the adaptation problem, but there is also a need to find the knowledge needed to create an original solution. Some mechanism must exist that will enable the creative problem solver to find necessary knowledge, even when it is conceptually distant from the original problem.

Creativity must be capable of "leaps" to new problem solutions.

The challenge of creativity research is to find a cognitive mechanism that is not crippled by these requirements. The creative process must be able to (1) find useful knowledge by searching the problem space, (2) limit the adaptation task, and (3) discover solutions substantially different from the original solution.

2.6 MINSTREL's Creativity Heuristics

The MINSTREL model of creativity is based on creativity heuristics that associate problem transformations with specific solution adaptations. To search the problem space for possible new solutions, MINSTREL begins at the original problem and applies small transformations that create new, slightly different problems. If one of these new problems can be solved, the associated solution adaptation can be used to create a new solution to the original problem.

Suppose, for example, that MINSTREL is trying to invent a method for a knight to kill a dragon. Figure 2.2 illustrates the search space for this problem. At the center is the original problem, and around it are similar problems. To discover a new solution to the original problem, MINSTREL applies a problem transformation, jumping from the original problems to one of the nearby problems. MINSTREL now tries to solve this new problem. If MINSTREL is suc-

![Figure 2.2 Search Space of Slightly Different Problems](image-url)
cessful (by recalling a solution from episodic memory), it then tries to adapt the recalled solution to the original problem, jumping back to the center of the search space. The result is a new solution to the original problem.

The process of “Transform-Recall-Adapt” is the heart of MINSTREL’s model of creative problem solving. In the Transform step, MINSTREL takes a problem description and changes it into a slightly different problem. In the Recall step, MINSTREL takes the new problem description and tries to recall similar past situations from episodic memory, and the solutions used in those situations. In the Adapt step, MINSTREL takes the recalled solutions and adapts them to the original problem. The Transform step searches the problem space, the Recall step finds new knowledge, and the Adapt step applies that knowledge to the current problem.

MINSTREL implements this process in creativity heuristics called Transform-Recall-Adapt Methods, or TRAMs. Each TRAM consists of a transformation to be applied to a problem and a specific adaptation that will apply recalled solutions to the original problem. Figure 2.3 shows a simplified example of one of MINSTREL’s TRAMS, TRAM: Cross-Domain-Solution. This TRAM suggests that a new solution to a problem can be found by (1) translating the problem into a new domain [Transform], (2) solving the problem in that domain [Recall], and (3) translating the solution back into the original domain [Adapt]. MINSTREL uses TRAM: Cross-Domain-Solution to create the following story scene (from “Richard and Lancelot”):

One day while out riding, Lancelot’s horse went into the woods. Lancelot could not control the horse. The horse took him deeper into the woods. The horse stopped. Lancelot saw Andrea, a Lady of the Court, who was out picking berries.

The original specification for this problem is “create a scene in which a knight accidently meets a princess.” TRAM: Cross-Domain-Solution transforms

**TRAM: Cross-Domain-Solution**

**Transform Strategy**

Find a new problem domain with similar actions and actors, and map the original problem into this new domain. Retain the mapping for use in the Adapt step.

**Adapt Strategy**

Map any discovered solutions back to the original domain by reversing the mapping used in the Adapt step.

Figure 2.3 TRAM: Cross-Domain-Solution

this problem specification by mapping it into the modern domain. Elements of the original problem specification are mapped to corresponding elements in the new domain. “A knight” becomes “a businessman” and “a princess” becomes “a friend.” The new problem specification is “a businessman accidently meets a friend” and recalls this story:

**Walking The Dog**

John was sitting at home one evening when his dog began acting strange. The dog was scratching at the door and whining for a walk. Finally, John decided to take the dog for a walk. While they were out, John ran across his old friend Pete, whom he hadn’t seen in many years.

This story is adapted to the original problem by mapping the story back into the King Arthur domain, creating the scene in which Lancelot’s horse leads him to Andrea. The resulting scene is creative—novel and useful—because TRAM: Cross-Domain enabled the problem solver to (1) discover knowledge previously unconnected to the original problem, and (2) apply that knowledge to create a new solution.

### 2.7 The MINSTREL Model of Creative Problem Solving

Figure 2.4 illustrates how TRAMs are integrated into the case-based problem solving model. The Recall and Adapt steps of the basic problem solving model are augmented with a pool of TRAMs. During problem solving, a TRAM is selected from this pool and applied to the original problem. If the TRAM succeeds in discovering a solution and adapting it to the original problem, then problem solving succeeds. If problem solving fails, then the current TRAM is discarded, another selected from the pool of available TRAMs, and the cycle repeats.

#### 2.7.1 Transform

The first step of the augmented problem solving model is to transform the original problem. In this step, the Transform portion of one of MINSTREL’s TRAMs is applied to the original problem description, changing it into a new problem. This step is a new addition to the model of case-based problem solving. It is this transformation that permits MINSTREL to search the problem space for new knowledge, a capability that normal problem solving lacks.

Of course, a problem solver needs to be creative only if problem solving fails. So before applying a transformation to the original problem specification, the problem solver should try to solve the original problem.
2.7.2 TRAM Selection

When TRAM:Standard-Problem-Solving fails to find a solution, a new TRAM must be selected from the pool of available TRAMs to begin MINSTREL’s effort to create a new solution.

The TRAMs in the pool of available TRAMs are organized according to the types of problems to which they apply. Some TRAMs are general, and can modify any problem description. Other TRAMs are specific to certain types of problem descriptions. To select a TRAM for the current problem, MINSTREL uses the problem specification to select the applicable TRAMs from the pool of available TRAMs.

MINSTREL indexes the pool of available TRAMs using episodic memory. Just as episodic memory contains past problems and their associated solutions, episodic memory also contains past problems and associated TRAMs. Finding the TRAMs that apply to a particular problem thus becomes a matter of recall. MINSTREL uses the current problem as an index to episodic memory, and finds similar past problems and their associated TRAMs. TRAMs that apply to many different types of problems are associated with very general problem descriptions; TRAMs that apply to specific types of problems are associated with more specific problem descriptions. By recalling all of the problem descriptions that match the current problem, MINSTREL gathers all the applicable TRAMs.

From the applicable TRAMs, one TRAM is selected randomly. The reason for using random choice is straightforward. Creativity heuristics search the problem space for useful knowledge to apply to the current problem. But prior to actually performing that search, the problem solver cannot know where the useful knowledge lies. In this sense, the applicable TRAMs are indistinguishable. Hence the problem solver has no reason to prefer one heuristic over another, making random selection a reasonable algorithm.

Another possible algorithm for selecting between competing TRAMs is to use past creativity experience to guide TRAM selection. For example, the problem solver might want to use TRAMs according to how frequently they’ve been successful in past problem solving situations. Or, since that might lead to repetition, the problem solver might want to try heuristics that haven’t been frequently used. Or the problem solver might want to try the most recently successful heuristic again, to reinforce the value of newly learned heuristics. All of these are interesting strategies.

Although MINSTREL does not currently use past creativity experience to guide TRAM selection, the use of episodic memory to organize TRAMs provides support for these types of strategies. As TRAMs are used and indexed into episodic memory, the problem solver retains knowledge about the usage of creativity. However, what form a cognitively valid record of creativity would take remains an open question.
2.7.3 Recall

The second step in the MINSTREL model of creative problem solving is recall of similar past problems. If similar past problems can be recalled, the associated solutions are passed on to the adaptation step. The index for recall is the problem specification. For this reason, episodic memory must be organized by past problems, and the recall process must be able to take a possibly incomplete problem specification and recall similar past problems.

MINSTREL's model of episodic memory is based on the "context plus index" model (Kolodner, 1984; Reiser, 1986; Schank, 1982). Context plus index models of memory organize a specific episode according to its general context (i.e., a plane trip) and the distinguishing features of the episode (i.e., taking Pan-Am, flying to Central America). In the MINSTREL model, problem types correspond to different contexts, and the specific features of each problem are used as indexing features. This permits MINSTREL to organize memory according to past problems, and to recall matching problems when given a problem specification.

2.7.4 Adaptation

The third step of MINSTREL's creative process is adaptation. Past solutions to problems are passed to the adaptation step, where the Adapt portion of the controlling TRAM modifies them for use on the current problem.

In general, the problem of adapting knowledge—even useful knowledge—to a new problem is very difficult. Consider what a problem solver must do to adapt the story "Walking the Dog" to the problem of "Lancelot meets Guinevere unexpectedly."

Walking The Dog

John was sitting at home one evening when his dog began acting strange. The dog was scratching at the door and whining for a walk. Finally, John decided to take the dog for a walk. While they were out, John ran across his old friend Pete, whom he hadn't seen in many years.

The problem solver must first fully understand this story, so that he can recognize that it is also an unexpected meeting. Then he must determine what its relationship is to the original problem, to know how to apply this knowledge to it. He must realize that it is in another problem domain, and determine what that domain is. Finally, he must build a mapping from this domain to the original problem domain and translate the story.

In MINSTREL, though, the problem of adaptation is greatly simplified by associating adaptations with specific problem transformations. Because each adaptation is applied only to problem solutions that arose from a particular problem transformation, there is no need to fully understand the problem solution and determine its relation to the original problem. The relation of the recalled solution to the original problem is fixed by the Transform portion of the TRAM. Instead, the adapt portion of each TRAM needs only do the final work of adaptation—to make the necessary changes to the recalled solution.

For example, in TRAM:Cross-Domain-Solution the Adapt step need only apply the cross domain mapping generated during the Transform step in reverse upon the problem solution. This translates the recalled solution back into the original problem domain, making it suitable for the original problem. The Adapt step does have to understand the recalled solution, or determine its relationship to the original problem.

2.7.5 Assessment

The fourth step of MINSTREL's creative process is assessment. The purpose of the assessment step is to evaluate a proposed solution in light of acceptance criteria specific to a particular domain. For example, a mechanical engineer might evaluate his designs in terms of their efficiency, what kinds of materials they use, and so on. Although the problem solving process tries to create solutions that fit the original problem specification, it may inadvertently fail (see the discussion of "Creativity Errors" in this chapter), or there may be additional criteria that cannot be easily expressed in the problem specification. In these cases, domain assessments can be used to catch faulty solutions.

A special assessment for artistic problem domains is the boredom assessment. The boredom assessment rejects solutions that are too similar to previous solutions, and models the drive for originality in the arts. The boredom assessment is an example of an assessment that embodies a criteria that is difficult to express in the original problem specification—namely, that the solution be original.

When a proposed solution fails a domain assessment, problem solving fails, the active TRAM and the proposed solution are discarded, and another TRAM is selected. The problem solving process then repeats under the control of the new TRAM.

One shortcoming of this algorithm is that it discards the proposed solution, which may be almost entirely correct. Re-inventing the correct parts of that solution may take a great deal of effort. An alternative possibility is to "repair" proposed solutions which fail a domain assessment.

For example, if a domain assessment for mechanical invention notices that a device has a redundant component, it can repair that design by removing the redundant component. Another possibility is to use problem solving recursively to repair a faulty solution. If a domain assessment in mechanical invention
notices that a device lacks a power source, it can repair this problem by recursively using problem solving to invent a power source.

MINSTREL is capable of both these types of repair. Domain assessments can directly manipulate the proposed solutions or call problem solving to create a correction. Whether a proposed solution should be rejected or repaired depends both upon the cost and efficacy of the repair technique.

First, repair should be undertaken only when it provides a cost savings over rejecting the faulty solution and finding a new solution. Repairing a solution with major faults may be much more expensive and time-consuming than simply throwing out the faulty solution and finding an entirely new one. For a mechanical device lacking a power source, inventing a power source may be a very difficult problem that would take more time and effort than simply finding a different solution to the original problem. If this is the case, it would be better to reject the proposed solution rather than repair it.

Efficacy of repair can also be a factor. Consider the earlier example in which a repair removes a redundant piece from a mechanical device. Is that always a correct and safe change to the design? Only insofar as the repair understands the device design. Perhaps the redundant pieces were added to the design by problem solving in order to balance the weight of the device. In general, a repair heuristic will not be as knowledgeable and powerful as problem solving, and so will sometimes make errors.

Deciding the likely costs of repair vs. rejection, or determining whether a repair will be efficient without introducing more serious faults are difficult issues. Currently MINSTREL's boredom assessment uses rejection, while the two assessments used in mechanical invention use repair, but these are simply design choices that are not founded on any deep understanding of the issues of repair vs. rejection. For further discussion of the role of repair in planning, see Hammond (1988).

2.8 Summary of Creativity Model

Earlier we identified the three challenges of creativity as: (1) finding useful knowledge by searching the problem space, (2) limiting the adaptation task, and (3) discovering solutions substantially different from the original solution. The MINSTREL model of creativity answers these challenges by associating problem transformations with corresponding solution adaptations. Problem transformations permit MINSTREL to search the problem space, and because each problem transformation has an associated, specific solution adaptation, the complexity of adaptation is eliminated. And this adaptation of knowledge from other problems leads to the creation of new solutions with substantial differences from previous solutions.

2.9 Leaps of Creativity

TRAMs find new solutions to problems by making small changes in the problem description and corresponding small adaptations to any discovered solutions. In light of the definition of creativity as a solution with a "substantial" difference from past solutions, this may seem counterintuitive. Why not use creativity heuristics that make large changes to the problem descriptions? There are several reasons, which are found in our current understanding of human cognition.

First, MINSTREL is an integrated model of problem solving and creativity, in which creativity is an extension of problem solving. In most problem solving situations, there is no need for powerful creativity. Most problems are solved using standard solutions or past solutions that have been only slightly adapted. It is only in rare cases that a problem solver must invent a solution substantially different from a past solution.

Consequently we expect creativity heuristics to be very efficient at discovering small adaptations while still capable of larger adaptations. By using heuristics that make small adaptations, MINSTREL is efficient at the types of simple problem solving and creativity that make up the bulk of problem solving situations. (The question of whether MINSTREL is capable of larger adaptations is discussed below.) But there are other reasons to use small adaptations as a basis for creativity.

One advantage of creativity heuristics that make only small changes to a problem description is that they are more likely to find useful knowledge. Slightly different problems are good sources of useful knowledge because they share many of the same constraints as the original problem, and their solutions are likely to have some applicability to the original problem. Examining slightly different problems constrains the search task to an area of the problem space that is localized and fertile.

A second advantage is that small adaptations are easier and more likely to be successful than large adaptations. Adapting a very different solution to a new problem (i.e., adapting the "getting to school" solution to the "spilled milk" problem) requires a great deal of knowledge and effort and is likely to fail no matter what the expenditure. Adapting a solution that has only small differences (i.e., adapting the "spilled juice" solution to the "spilled milk" problem) requires less knowledge and effort and is more likely to succeed.

So there are several reasons to use small adaptations. But are small adaptations capable of discovering more creative solutions?

Sometimes even simple heuristics are capable of discovering unique solutions. Consider the heuristic to "substitute an agent" which Janelle used to create a solution involving a kitten. Although the heuristic itself is simple and commonplace, in this case the solution it discovers is surprisingly creative.

But even if creativity heuristics taken singly are not capable of major
discoveries, they can be when taken in combination. The strategy of small problem transformations tends to find solutions with small differences from the original problem. But if a single heuristic does not find a new solution, transformations can be repeatedly applied until substantially different solutions are discovered. And because the adaptation of the created solution is done in many small, simple steps, the process of adaptation remains simple.

Leaps in creativity result from combinations of small modifications.

How can several creativity heuristics be applied to a single problem? In MINSTREL, this is achieved through a mechanism called imaginative memory.

2.9.1 Imaginative Memory

The central step of MINSTREL's creative problem solving model is recalling a solution from episodic memory. But recall itself can be viewed as a kind of "problem solving." What happens if creative problem solving is used to solve the "recall problem"?

To do this requires replacing the Recall step of creative problem solving with a recursive call to creative problem solving with the problem specification "Find something in episodic memory that matches these features." Of course, this could lead to endless recursion: at each level of problem solving, the Recall step calls problem solving again. To terminate this endless recursion, TRAM:Standard-Problem-Solving is modified so that it will continue to use episodic memory for recall. Since TRAM:Standard-Problem-Solving is always the first TRAM used by creative problem solving, this means that the first attempt at recall at each level of recursion will use episodic memory instead of recursing to another level of problem solving. The result is that creative problem solving first tries to recall a solution from episodic memory. If that fails, it recursively calls creative problem solving to solve the "recall" problem. This process is illustrated in Figure 2.6.

Now when a problem solver needs to recall something, TRAM:Standard-Problem-Solving is the first TRAM used, and passes the recall features unchanged to episodic memory. If an episode that matches the recall features is found, problem solving succeeds. Because TRAM:Standard-Problem-Solving is always the first TRAM used and continues to use episodic memory normally, recall behaves as expected when an episode exists that matches the recall features.

Something more interesting happens when the Recall step of TRAM:Standard-Problem-Solving fails. If TRAM:Standard-Problem-Solving cannot find an episode in memory that matches the recall features, problem solving fails and a new TRAM is selected. This TRAM modifies the recall features and recursively calls the problem solving process with the new recall features.

![Figure 2.6 Recursive Creativity](image_url)

The first TRAM used on the recursive call is TRAM:Standard-Problem-Solving. If the new features recall an episode, the episode is returned to the previous recursion of problem solving, where it is adapted to the original problem by the previous TRAM, and recall succeeds. But because the recalled episode was changed by the Adapt portion of the previous TRAM, it is no longer the episode that was found in memory.

Recall has succeeded in a strange way: by recalling an episode that does not exist in episodic memory. Episodic memory has become imaginative. When an appropriate episode exists, it is recalled. When no appropriate episode exists, recall uses creativity heuristics to "imagine" an appropriate episode.

Treating recall as problem solving also enables the problem solver to apply multiple TRAMs to a problem. Each time recall fails the recursive use of creative problem solving will apply another TRAM to the recall features. In this
way, a number of TRAMs can be successively applied to a problem. Each TRAM changes the problem in only a small way, but the cumulative effect may be large, enabling the creative problem solver to discover new solutions significantly different from known solutions.

There are two advantages to imaginative memory.

First, it provides a simple and powerful mechanism for the repeated applications of creativity heuristics to a problem. Each time the Recall step of problem solving fails, imaginative memory will recurse and apply a new creativity heuristic. If no useful knowledge can be found in the problem space near the original problem, repeated problem transformations will move the creative problem solver into more distant areas of the problem space.

More importantly, imaginative memory implements creativity at a low cognitive level. By embedding creative problem solving in the recall process, imaginative memory makes creativity transparently available to any cognitive mechanism that uses episodic memory for reasoning. By integrating creativity into the foundation of the cognitive process, imaginative memory increases the reasoning power of all cognitive mechanisms.

2.10 Integrated Model

Figure 2.7 illustrates how the boredom assessment, Transform-Recall-Adapt Methods and imaginative memory are integrated with case-based reasoning to form a complete model of creative problem solving. The three steps of case-based reasoning (Recall, Adapt, Assess) have been augmented with a Transform step. An active TRAM controls the Transform and Adapt steps. Initially this is TRAM:Standard-Problem Solving, which is simply the strategy of recalling a similar past problem and using the solution from that problem. The Assess step applies a pool of assessments to proposed solutions. In creative domains, this includes the boredom assessment, which rejects solutions that are too similar to past solutions. The Recall step uses imaginative memory (a recursive call to problem solving) except when controlled by TRAM:Standard-Problem Solving.

The problem solving cycle begins when a problem description enters the recall step. Initially TRAM:Standard-Problem Solving is in control. The original problem description is used to recall similar problem solving situations from episodic memory. If recall succeeds, the recalled situations are passed to the Adapt step. Under TRAM:Standard-Problem Solving, no adaptation is needed because the recalled solutions are very similar to the original problem, so the recalled solutions are passed along to the Assess step. In the Assess step, all active assessments are applied to the recalled solutions, and if a solution passes all the assessments, it is output as a solution to the original problem. This is the normal problem solving cycle.

If TRAM:Standard-Problem Solving fails, either because no solutions were recalled or because the recalled solutions failed some assessment,
2.11 Examples of Creativity

The model of creative problem solving presented in this chapter has been implemented in a computer program called MINSTREL. The next two sections show how MINSTREL uses creative problem solving to discover new solutions for a simple planning problem in the domain of King Arthur, and to create a scene for a story. These examples should give the reader a better understanding of the TRAM model of creativity.

2.11.1 Suicide Example

In this example, MINSTREL is trying to discover a way for a knight to commit suicide. Initially, MINSTREL knows nothing about suicide, but does know about killing dragons and drinking a potion to become ill. Using this knowledge and creative problem solving, MINSTREL discovers three methods of suicide and invents the notion of “poison.”

2.11.1.1 Representation

MINSTREL uses a schema-based representation language called RHAPSODY Turner (1985). Goals, actions, and states of the world are represented as schemas; instances of these schemas make up the episodes in MINSTREL’s memory and the elements of the stories MINSTREL tells. Each schema has named slots which contain schema information. Goal schemas, for example, have slots for the type of the goal and the actor of the goal. Schemas can also have named links to other schemas. Goal schemas typically have links to plans, and to the states that achieve the goals. Schema names begin with an ampersand (&) and schema instances are given either descriptive names (such as &KNIGHT-FIGHT) or generated names based on the schema type (such as &GOAL.112). For a more complete discussion of MINSTREL’s representation, see (Turner, 1985).

2.11.1.2 The Problem

Figure 2.8 illustrates MINSTREL’s representation of the suicide problem. &HUMAN.12 is an instance of the human schema which represents the knight. The type slot of a human instance indicates the character’s major role in the King Arthur world, and illustrates how MINSTREL uses schema slots to instantiate particular schema instances. The knight has a goal (&GOAL.11) which will be achieved by the knight being dead (&STATE.8). The plan for this goal (&ACT.4) is currently uninstantiated. MINSTREL's goal in this example is to instantiate &ACT.4 as an action or series of actions that will achieve the knight’s goal of committing suicide.

2.11.1.3 Initial State of Episodic Memory

All of MINSTREL’s knowledge of the King Arthur domain is contained in episodic memory. MINSTREL’s creativity heuristics have general knowledge about goals, plans, and states of the world, but specific knowledge about the goals, plans, and actions of characters in the King Arthur domain is deduced from the contents of episodic memory.

At the beginning of this example, MINSTREL knows nothing about how a knight might kill himself. Initially, MINSTREL’s episodic memory contains only these two episodes:

**Knight Fight**

A knight fights a troll with his sword, killing the troll and being injured in the process.

**The Princess and the Potion**

A lady of the court drank a potion to make herself ill.

![Figure 2.8 Representation of Suicide](image-url)

---

4 I use the term *invent* to indicate a concept that is new to MINSTREL, if not necessarily new to the reader.
Figure 2.9 shows the schema representation of "The Princess and the Potion." \&ACT.14 represents the action of Lady Andrea quaffing a potion. \&STATE.17 represents the intentional outcome of that action—Lady Andrea becoming ill.

2.11.1.4 Trace

Figure 2.10 shows a trace of MINSTREL inventing three different methods of suicide. In this example, MINSTREL has been configured to exhaustively invent solutions to the suicide problem and present them in English as created. Normally MINSTREL generates copious debugging and tracing output. To spare the reader, the trace shown in Figure 2.10 has been edited to improve readability. Uninteresting portions of the trace and reasoning dead-ends have been deleted. These deletions have been marked in the trace with "[...]." Except for this editing, the trace appears exactly as generated by MINSTREL. The level of indentation of the trace reflects the level of recursive problem solving.

MINSTREL begins the example shown in Figure 2.10 by generating the initial problem specification in English: "A knight named John did something. John died." This is an English description of the schema representation shown in Figure 2.7. MINSTREL’s English descriptions of schemas are produced by a phrasal generator (Reeves, 1989; Zernik, 1987).

The problem specification is followed by a trace of the problem solving cycle. As each new TRAM is applied, the name of the TRAM (i.e., TRAM:GENERALIZE-CONSTRAINT) is printed out. When TRAM:Standard-Problem Solving is used to attempt recall from episodic memory, a message is printed out showing the recall index and what was recalled. The very first part of the trace shows TRAM:EXAGGERATE-SCALED-VALUE being applied to this problem and failing when nothing (i.e., NIL) is recalled from episodic memory.

**MINSTREL Invention**

**Problem Solving Cycle:** \&ACT.105.

Executing TRAM:EXAGGERATE-SCALED-VALUE.

Recalling \&ACT.105: NIL.

...TRAM fails.

[...]

Executing TRAM:GENERALIZE-CONSTRAINT.

Generalizing :OBJECT on \&STATE.112.

Recalling \&ACT.118: \&KNIGHT-FIGHT.

TRAM succeeds: (\&ACT.405).

Minstrel invented this solution:

(A KNIGHT NAMED JOHN FOUGHT HIMSELF BY MOVING HIS SWORD TO HIMSELF IN ORDER TO KILL HIMSELF *PERIOD* JOHN DIED *PERIOD*)

[...]

Executing TRAM:SIMILAR-OUTCOMES-PARTIAL-CHANGE.

Recalling \&ACT.136: NIL.

[TRAM Recursion: \&ACT.136.]

Executing TRAM:GENERALIZE-CONSTRAINT.

Generalizing :ACTOR on \&ACT.138.

Recalling \&ACT.138: \&PRINCESS-POTION.

...TRAM succeeds: (\&ACT.447).

...TRAM succeeds: (\&ACT.447).

Minstrel invented this solution:

(A KNIGHT NAMED JOHN DRANK A POISON IN ORDER TO KILL HIMSELF *PERIOD* JOHN DIED *PERIOD*)

[...]

Executing TRAM:INTENTION-SWITCH.

Recalling \&ACT.174: NIL.

[TRAM Recursion: \&ACT.174.]

Executing TRAM:SIMILAR-OUTCOMES-PARTIAL-CHANGE.

Recalling \&ACT.178: \&KNIGHT-FIGHT.

...TRAM succeeds: (\&ACT.588).

...TRAM succeeds: (\&ACT.588).

Minstrel invented this solution:

(A KNIGHT NAMED JOHN FOUGHT A DRAGON BY MOVING HIS SWORD TO IT IN ORDER TO KILL HIMSELF *PERIOD* JOHN DIED *PERIOD*)
When a solution is discovered, MINSTREL prints a message to that effect and generates an English language description of the solution.

2.11.1.5 TRAM:Generalize-Constraint

The first TRAM that succeeds in discovering a solution to the suicide problem is TRAM:Generalize-Constraint. This TRAM suggests that a new solution to a problem can be found by removing a solution constraint, solving the new problem, and then adding the constraint back to the new solution. Figure 2.11 shows an informal outline of this heuristic.

In the suicide problem, the constraints available for generalization are the schema slot fillers of the problem specification (Figure 2.7): (1) the actor is a knight, (2) the object of the state is the actor, (3) the type of the state is health, and (4) the value of the state is dead. TRAM:Generalize-Constraint suggests recalling scenes in which one of these constraints has been generalized.

In this example, MINSTREL generalizes constraint (2). The original problem specification is “a knight kills himself.” TRAM:Generalize-Constraint generalizes this specification by removing the constraint that the knight kill himself and replacing it with the more general constraint that the knight kill something. This is indicated by the message “Generalizing :OBJECT on &STATE.112.” printed in the trace. The new problem specification is “a knight kills something.” This generalization recalls the “Knight Fight” episode:

\[
\begin{align*}
\text{TRAM:Generalize-Constraint} \\
\text{Transform} \\
\text{1. Select and generalize a feature (call it $\text{generalized-feature}$) of the scene specification. Use this new scene specification as an index for imaginative recall.}
\end{align*}
\]

Adapt

1. Adapt the recalled solution to the current problem by adding $\text{generalized-feature}$ back to the recalled scene.

Figure 2.11 Informal Outline of TRAM:Generalize-Constraint

[...] Executing TRAM:GENERALIZE-CONSTRAINT.
Generalizing :OBJECT on &STATE.112.
Recalling &ACT.118: &KNIGHT-FIGHT.
Adapting by replacing &MONSTER.15 with &HUMAN.12.
TRAM succeeds: (&ACT.405). 

In “Knight Fight,” a knight kills a troll by hitting it with his sword. This episode is adapted to the original suicide problem by reversing the original transformation. The troll corresponds to the generalized constraint that a knight kills “something.” To adapt this solution to the original problem, this more general constraint must be replaced with the original constraint—that the knight kill himself. Therefore the Adapt portion of TRAM:Generalize-Constraint replaces the troll with the knight, creating a scene in which a knight kills himself by hitting himself with his sword:

Minstrel invented this solution: 
(A KNIGHT NAMED JOHN FOUGHT HIMSELF BY MOVING HIS SWORD TO HIMSELF IN ORDER TO KILL HIMSELF *PERIOD* JOHN DIED *PERIOD*)

TRAM:Generalize-Constraint has used previous knowledge about how knights kill monsters to create a scene in which a knight kills himself. This process is shown graphically in Figure 2.12.

Three issues that must be addressed in TRAM:Generalize-Constraint are (1) how features are selected for generalization, (2) how the selected feature is generalized, and (3) how the selected feature is added back into the created scene.

To maximize the success of the recall step of TRAM:Generalize-Constraint, the feature chosen for generalization should be likely to lead to the recall of a scene. To achieve this, MINSTREL makes a broad generalization of each feature in the representation and attempts recall using the generalized episode. Every generalization that results in recall is added to a pool, and the problem constraint to be generalized is selected randomly from this pool.

MINSTREL uses two methods to generalize a feature. First, the feature can be completely removed from the problem specification. This is the broadest possible generalization, and is used when selecting the candidate pool. And while this method provides a good, quick indication of whether generalizing a particular feature is useful, it is so broad that it often leads to the recall of scenes which are difficult to adapt to the original problem.

For example, suppose that MINSTREL is creating a scene in which “a knight gives a princess something that makes her happy” and chooses to generalize the “princess” feature by removing it altogether from the problem specification. The new problem specification—“A knight gives somebody something that makes him happy”—can recall any episode in memory in which a knight
feature and the instantiations of the generalization. One such generalization is based on class hierarchies.

Classes group concepts that have many similarities. The “People” class groups a variety of characters—princesses, knights, kings, and hermits—that have many similar features. There can be many class hierarchies, and objects can belong to several classes. Knights, for example, are members of both the “People” class and the “Violent Characters” class. By generalizing problem features within classes, MINSTREL is more likely to find a useful reminding.

For this reason, MINSTREL’s implementation of TRAM:Generalize-Constraint uses class generalizations. In the suicide example, the “knight” feature is generalized to “a Violent Character.” This recalls the “Knight Fight” episode, in which a knight fights a troll in order to kill the troll, because trolls are also members of the “Violent Character” class. If this generalization had failed, MINSTREL would have generalized to the superclass (“Actors”), and if that generalization failed, TRAM:Generalize-Constraint would have failed. A portion of MINSTREL’s class hierarchy is shown in Figure 2.13.

The final step in TRAM:Generalize-Constraint is to adapt the recalled episode to the original specification. This is achieved by replacing the generalized feature value with the original feature value throughout the recalled episode. In the suicide example, the troll in “Knight Fight” is replaced by the knight throughout the recalled episode, resulting in a scene in which a knight kills himself by fighting himself with his sword.

Figure 2.14 illustrates TRAM:Generalize-Constraint as implemented in the current version of MINSTREL.

2.11.1.6 TRAM:Similar-Outcomes

MINSTREL discovers a second method for committing suicide by using both TRAM:Generalize-Constraint and a new heuristic, TRAM:Similar-Outcomes-Partial-Change. TRAM:Similar-Outcomes-Partial-Change suggests that if an action results in a particular outcome, it might also result in other, similar outcomes. For example, if MINSTREL doesn’t know anything about riding horses except that a knight once rode one to a castle, MINSTREL can use TRAM:Similar-Outcomes to guess that a knight might also ride a horse to some other destination.

In the suicide example, TRAM:Similar-Outcomes-Partial-Change recognizes that being killed is similar to being injured, and transforms the problem description from “a knight purposely kills himself” to “a knight purposely injures himself.” If MINSTREL can recall an action in which a knight purposely injures himself, it will be adapted to the current problem by replacing the injury with death. In essence, TRAM:Similar-Outcomes-Partial-Change “guesses” that an
action that is known to result in injury might also result in death. MINSTREL
then tries to recall a scene in which “a knight purposely injures himself”:

[...]  
Executing TRAM:SIMILAR-OUTCOMES-PARTIAL-CHANGE.  
Transforming &DEATH to &WOUND.  
Recalling &ACT.136: NIL.

However, this does not recall either of the episodes in MINSTREL's episodic
memory (as indicated by "Recalling &ACT.136: NIL."). "Knight
Fight" is not recalled because the knight does not intentionally injure himself,

TRAM: Generalize-Constraint

Transform Strategy

1. For each feature in the scene specification, eliminate the feature and
attempt recall from episodic memory. If recall is successful, then add
the feature to a pool of acceptable generalizations.

2. Randomly choose a feature from the pool of acceptable
generalizations ($feature).

3. Create a generalization of $feature based on the class membership.
Attempt to recall or create a scene based on that generalization. If
successful, pass the recalled scene ($recall) to the adapt step.

4. Otherwise, create a generalization based on the superclass, and
attempt recall.

5. Otherwise, create a generalization by eliminating the feature, and
repeat, and attempt recall.

6. Otherwise, fail this feature and attempt another feature from the pool
of possible generalizations.

Adapt Strategy

1. Build a correspondence between the original scene specification
($original) and $recall by matching all the features in $original with the
same features in $recall.

2. Add to this correspondence a mapping from the selected feature
($feature) to the instantiation of that feature in $recall.

3. Copy features that are present in $recall but missing in $original from
$recall to $original, translating through the correspondence.

"Princess and the Potion" is not recalled because the actor is not a knight.
Since recall fails, imaginative memory recurses and applies a new TRAM:
Executing TRAM:SIMILAR-OUTCOMES-PARTIAL-CHANGE.

Transforming &DEATH to &WOUND.
Recalling &ACT.136: NIL.

[TRAM Recursion: &ACT.136.]
Executing TRAM:GENERALIZE-CONSTRAINT.
Generalizing :ACTOR on &ACT.136.
Recalling: &PRINCESS-POTION.
...TRAM succeeds: (&ACT.447).
...TRAM succeeds: (&ACT.447).

Minstrel invented this solution:
(A KNIGHT NAMED JOHN DRANK A POISON IN ORDER
TO KILL HIMSELF *PERIOD* JOHN DIED *PERIOD*)

At this new level, MINSTREL applies TRAM:Generalize-Constraint to the description “a knight purposely injures himself” and generalizes the “knight” feature. “Knight” is generalized to “anyone,” which results in the new problem specification “Someone does something to purposely injure himself.” Note that this problem specification has been modified twice from the original specification, once by TRAM:Similar-Outcomes-Partial-Change, and once by TRAM:Generalize-Constraint.

The new problem description recalls “The Princess and the Potion” in which a lady of the court drinks a potion to make herself ill. The Adapt portion of TRAM:Generalize-Constraint then adapts this scene by replacing “lady of the court” (the generalized constraint) with “a knight” (the original constraint). This results in a scene in which a knight makes himself ill by drinking a potion.

The adapted scene is returned to the previous problem solving level, where TRAM:Similar-Outcomes-Partial-Change also adapts the scene, by replacing the illness with death. The adaptation reverses the transformation that turned “death” into “illness.” This results in a scene in which a knight kills himself by drinking a potion, filling the original description “a knight kills himself.” Note that in the course of inventing this scene, MINSTREL has also invented the idea of poison—a potion that kills (vs. just making one ill).

(In fact, MINSTREL has invented the more narrow notion of “a potion which will kill a knight.” To apply this to other animate beings, MINSTREL will have to use creativity again to generalize about the actor of this action. MINSTREL does this by applying TRAM:Generalize-Constraint again.)

The main issue in TRAM:Similar-Outcomes is determining when two outcomes are interchangeable. MINSTREL has two methods for deciding this question. These are implemented as separate TRAMs called TRAM:Similar-Outcomes-Partial-Change and TRAM:Similar-Outcomes-Implicit.

TRAM:Similar-Outcomes-Partial-Change reasons that if an action can result in a partial relative change of a state then the action can also result in a complete change of the state. TRAM:Similar-Outcomes-Partial-Change is shown in

Figure 2.15. In this example, TRAM:Similar-Outcomes-Partial-Change reasons that something that makes someone ill (a partial negative change in health) might also kill them (a complete negative change in health).

Like many of MINSTREL’s creativity heuristics, TRAM:Similar-Outcomes-Partial-Change is a “common-sense” rule that captures reasoning that is often useful but not always correct. For example, extending a partial state change could be used to reason that because a man can lift a book in one hand he would also be able to lift an automobile in one hand. There are two things to be said about this type of error.

First, this type of error points out the value of simple, constrained TRAMs that make only small changes in problem descriptions. By making only small extensions to a problem solver’s knowledge, a TRAM is less likely to make an error in its extrapolation. In fact, MINSTREL’s version of TRAM:Similar-Outcomes-Partial-Change only extends state changes one additional “step” in a known direction. If state of health is represented by the scale “Excellent Good Normal Ill Dead,” and MINSTREL knows an action that changed a man’s health from Normal to Ill, then TRAM:Similar-Outcomes-Partial-Change can only extrapolate that to an action that changes a man’s health from Normal to Dead. TRAM:Similar-Outcomes-Partial-Change cannot extrapolate to an action that would take a man’s health from Ill to Excellent, or even from Excellent to Dead. Similar limits can be applied to states which do not have discrete representations, although MINSTREL does not currently handle this. By using simple, constrained creativity heuristics, MINSTREL reduces the number of reasoning errors it makes.

Incremental imaginative steps reduces errors in reasoning.

Second, even with restricted TRAMs, MINSTREL can still make reasoning errors of this sort. But this is to be expected: A creative problem solver should make errors. MINSTREL uses creativity to actively extend its knowledge. By

TRAM:Similar-Outcomes-Partial-Change

Transform Strategy

If the problem specification has an act that results in a partial relative change of a state in some direction, create a new specification in which the relative change is extended in the same direction.

Adapt Strategy

Replace the change of state in the recalled episode with a relative change of state copied from the original problem specification.

Figure 2.15 TRAM:Similar-Outcomes-Partial-Change
making good use of what it already knows, MINSTREL can often make accurate
guesses about what it doesn’t know. But sometimes it will err. The challenge
faced by a creative problem solver is to find creativity heuristics that are produc-
tive without an inordinate number of reasoning errors. The issue of creativity
errors is discussed in more detail in Chapter 8.

Imaginative reasoning will sometimes produce errors.

The second TRAM MINSTREL uses to determine when two outcomes are
interchangeable is TRAM:Similar-Outcomes-Implicit. TRAM:Similar-
Outcomes-Implicit reasons that two outcomes are interchangeable in every situation
if it can recall any situation in which they are interchangeable. For example, if
MINSTREL can recall a scene in which a knight fought and killed a troll, and
another scene in which a knight fought and killed a dragon, MINSTREL can
use this knowledge to guess that trolls and dragons are generally inter-
changeable. TRAM:Generalize-Constraint used a class hierarchy—an explicit represen-
tation of object similarities—to substitute one feature for another. TRAM:
Similar-Outcomes uses an implicit representation of similarities to sub-
titute one outcome for another.

Like TRAM:Similar-Outcomes-Partial-Change, TRAM:Similar-Outcomes-
Implicit is a heuristics that can sometimes err. Again, this is a direct conse-
quence of its function as an extrapolator of knowledge, and is to be expected in a
creative problem solver.

TRAM:Similar-Outcomes-Implicit

Transform Strategy

1. Create a new problem specification: An uninstantiated act that results
in the state from the original problem specification. Use this to recall
different acts which can cause the result from the original problem.

2. Use episodic memory to recall other possible results of the actions
collected in (1). Build a pool of these alternate results.

3. Create a new problem specification in which the act from the original
problem specification results in a randomly-selected alternate result. Use
this new specification for recall.

Adapt Strategy

1. Replace the alternate result of the recalled episode with the result
copied from the original problem specification.

Figure 2.16 TRAM:Similar-Outcomes-Implicit

2.11.1.7 TRAM:Intention-Switch

MINSTREL’s final plan for suicide is discovered using TRAM:Intention-Switch.
This heuristic suggests that if the effect of an action was intentional it might just
as well have been unintentional. TRAM:Intention-Switch is illustrated in Figure
2.17.

In the suicide example, TRAM:Intention-Switch transforms the original
specification from “a knight purposely kills himself” to a “knight accidently
kills himself.” Recall on this new specification (&ACT.174) fails, because
MINSTREL’s episodic memory does not contain any episodes in which a knight
accidently kills himself:

[...]

Executing TRAM:INTENTION-SWITCH.
Recalling &ACT.174: NIL.

[TRAM Recursion: &ACT.174.]

Executing TRAM:SIMILAR-OUTCOMES-PARTIAL-CHANGE.
Recalling &ACT.178: &KNIGHT-FIGHT.

...TRAM succeeds: (&ACT.588).

...TRAM succeeds: (&ACT.588).

Martin invented this solution:

(A KNIGHT NAMED JOHN FOUGHT A DRAGON BY MOVING
HIS SWORD TO IT IN ORDER TO KILL HIMSELF
*PERIOD* JOHN DIED *PERIOD*)

Problem solving is used recursively, and TRAM:Similar-Outcomes-Partial-
Change modifies the current problem description “a knight accidently kills him-
self” by changing “kills himself” into something similar: “injures himself.”
The new problem description is “a knight “accidently injures himself.” This
recalls “Knight Fight”, in which a knight is injured while killing a troll.

TRAM:Intention-Switch

Transform Strategy

If an action in the problem specification intends a result, create a new
problem specification in which the same action unintentionally achieves
the result.

Adapt Strategy

Replace the unintentional result of the recalled episode with a similar
intentional result.

Figure 2.17 TRAM:Intention-Switch
2.11.2 Storytelling Example

We now look at how MINSTREL's creativity functions in the context of a larger task: storytelling.

To tell stories, MINSTREL must select a theme, instantiate the events of the theme (the plot), assure that the events of the story are consistent, achieve literary goals such as building suspense, and so on. Many of these tasks can be achieved without creativity. MINSTREL knows that knights ride horses, and hence doesn't have to invent a way for knights to travel from place to place. But sometimes MINSTREL encounters a new problem in the course of storytelling, or becomes bored with a particular story development. In these cases, creative problem solving is used to invent a solution.

This example presents a specific task MINSTREL encountered in telling The Vengeful Princess:  

The Vengeful Princess

Once upon a time there was a lady of the court named Jennifer. Jennifer loved a knight named Grunfeld. Grunfeld loved Jennifer.

Jennifer wanted revenge on a lady of the court named Darlene because she had the berries which she picked in the woods and Jennifer wanted to have the berries. Jennifer wanted to scare Darlene. Jennifer wanted a dragon to move towards Darlene so that Darlene believed it would eat her. Jennifer wanted to appear to be a dragon so that a dragon would move towards Darlene. Jennifer drank a magic potion. Jennifer transformed into a dragon. A dragon move towards Darlene. A dragon was near Darlene.

Grunfeld wanted to impress the king. Grunfeld wanted to move towards the woods so that he could fight a dragon. Grunfeld moved towards the woods. Grunfeld was near the woods. Grunfeld fought a dragon. The dragon died. The dragon was Jennifer. Jennifer wanted to live. Jennifer tried to drink a magic potion but failed. Grunfeld was filled with grief.

Jennifer was buried in the woods. Grunfeld became a hermit.

MORAL: Deception is a weapon difficult to aim.

Footnotes:
3Titles for MINSTREL's stories were provided by the author. Throughout this dissertation, MINSTREL's stories are presented exactly as produced, except for typography.
The particular portion of *The Vengeful Princess* this example focuses on is the creation of Jennifer's reason for wanting revenge on Darlene:

... Jennifer wanted revenge on a lady of the court named Darlene because Darlene had the berries which she picked in the woods and Jennifer wanted to have the berries.

When telling this story, MINSTREL knew nothing about what kinds of goal conflicts might lead a lady of the court to want revenge on another lady. This example shows how MINSTREL invents a conflict over possession of berries as a reason for wanting revenge.

2.11.2.1 The Problem

This example looks at how MINSTREL invents a reason for Jennifer to want revenge on Darlene. MINSTREL's representation of this problem is shown in Figure 2.18. Jennifer's goal of wanting revenge (&GOAL.1751) is motivated by a state of the world (&STATE.992) that achieves Darlene's goal (&GOAL.3029) at the expense of Jennifer's goal (&GOAL.2112). (One of the interesting storytelling aspects of this story is that MINSTREL knows that Jennifer wants revenge before it knows why Jennifer wants revenge. This is a consequence of how MINSTREL develops stories from the theme outward.)

When this example begins, MINSTREL has three goals: (1) to instantiate the state that causes Jennifer to want revenge (&STATE.992), (2) to instantiate Darlene's goal achieved by this state (&GOAL.3029), and (3) to instantiate Jennifer's thwarted goal (&GOAL.2112).

2.11.2.2 Episodic Memory

For storytelling, MINSTREL's episodic memory contains 10 story fragments from the King Arthur domain. For this example, the only relevant episode is "Picking Berries":

Picking Berries

A lady named Guinevere who wanted berries went to the woods and picked some.

\[ \text{Figure 2.18 Initial Problem Specification} \]

2.11.2.3 Standard Problem Solving During Storytelling

MINSTREL uses creative problem solving to instantiate Jennifer's thwarted goal (&GOAL.2112 in Figure 2.18). But before MINSTREL does this, it instantiates the state which thwarted this goal (&STATE.992 in Figure 2.18). To instantiate this state, MINSTREL uses standard case-based problem solving.

TRAM:Standard-Problem Solving is a TRAM that implements standard case-based problem solving. Given a problem description, TRAM:Standard-Problem Solving tries to recall from episodic memory an exactly similar problem. If it can, it uses the solution from that previous problem to solve the current problem. TRAM:Standard-Problem Solving is illustrated in Figure 2.5.

In this case, the problem description is &STATE.992: "Something happens which fulfills a princess's goal." Without transformation, this recalls a similar episode from memory: the "Picking Berries" story fragment. This scene is used to fill in as much of &STATE.992 and surrounding schemas as possible. The scene development at this point is shown in Figure 2.19. Notice that this has also resulted in the addition of a new schema to the story, &ACT.1178. This new schema represents Darlene's action in picking the berries.
This example illustrates how standard case-based problem solving occurs in MINSTREL. TRAM:Standard-Problem Solving recalls an episode from memory and applies it without modification to the current problem. What happens when TRAM:Standard-Problem Solving cannot recall an appropriate episode?

### 2.11.2.4 Creative Problem Solving During Storytelling

MINSTREL now tries to instantiate Jennifer’s thwarted goal. To do this, MINSTREL must show how Darlene’s possession of the berries could thwart one of Jennifer’s goals. MINSTREL’s author-level representation of this goal is &GOAL.3067, and Figure 2.20 shows a trace of MINSTREL achieving this goal.

The first part of this trace shows MINSTREL recalling author-level plans for instantiating a story scene. MINSTREL recalls 4 plans. The first, ALP:Don’t-Instantiate, fails. The second, ALP:General-Instantiate, succeeds. ALP:General-Instantiate tries to instantiate a story scene by using creative problem solving at the character level. The second portion of the trace (beginning with “TRAM Cycle: &GOAL.2112”) shows creative problem solving being used to instantiate the scene.

ALP:General-Instantiate passes the story scene to be instantiated to problem solving. TRAM:Standard-Problem Solving is tried but fails, because MINSTREL does not have any scenes in episodic memory in which a lady’s goal is thwarted by someone else possessing some berries. (In fact, MINSTREL’s memory does not contain any scenes in which a lady’s goal is thwarted.)

TRAM:Standard-Problem Solving fails, so it is discarded. The next TRAM used is TRAM:Opposite-State-Achieves. This TRAM suggests that the opposite of a state that achieves a goal will thwart the goal, and vice versa. If being healthy achieves the goal of protecting your health, then being dead will likely thwart the goal of protecting your health, and so on. So to invent a thwarted goal, you can recall an achieved goal and reverse it.

In this case, the opposite of the thwarting state (Darlene’s possession of the berries) is Darlene not possessing the berries (i.e., someone else possessing the berries). TRAM:Opposite-State-Achieves changes the problem specification from “A princess’s goal is thwarted by Darlene possessing some berries” to “A princess’s goal is achieved by someone possessing some berries.” If TRAM:Opposite-State-Achieves can recall something similar to this new specification, it can be used in the original problem by reversing the recalled scene from achievement to thwarting.

MINSTREL constructs this opposite state and tries to recall goals from episodic memory that are achieved by this new state. This recalls &GOAL-BERRIES, which is Guinevere’s goal of wanting to possess berries from the “Picking Berries” episode: “Guinevere’s goal of possessing berries is achieved by Guinevere possessing the berries.”

The recalled episode can be adapted to the original problem by filling Jennifer and Darlene into the correct roles and “reversing” the outcome. This is achieved in three steps: (1) replacing the actor of the goal with Jennifer (i.e., “Jennifer’s goal of possessing the berries is achieved by Guinevere possessing...
the berries”), (2) replacing the possessor of the berries with Darlene (i.e., “Jennifer’s goal of possessing the berries is achieved by Darlene possessing the berries”), and (3) by replacing the achievement with thwarting (i.e., “Jennifer’s goal of possessing the berries is thwarted by Darlene possessing the berries”).

The adapted solution can now be used to fill in the original scene. The result is shown in Figure 2.21. In English, the scene shown in Figure 2.21 is expressed:

... Jennifer wanted revenge on a lady of the court named Darlene because Darlene had the berries which she picked in the woods and Jennifer wanted to have the berries.

Notice what has happened during the creation of this thwarted goal. Prior to creating this goal, MINSTREL had no explicit knowledge about conflicts of possession, or of the idea that one person’s possession of an object prevents another person from also possessing the object. By using a very simple story episode and a general creativity heuristic, MINSTREL was able to invent these concepts and apply them to a specific problem. And as these concepts are invented they are indexed into episodic memory, where they are available for future problem solving, or as a basis for additional creativity. In this way, MINSTREL uses creativity to constantly expand its knowledge, and avoids having to reinvent the same concepts over and over.

2.11.2.5 Summary

This example demonstrates MINSTREL’s use of problem solving and creativity during storytelling. By using TRAM:Opposite-State-Achieves, MINSTREL solves a problem it cannot solve during standard problem solving. In solving this goal, MINSTREL discovers the idea of conflict over possession of an object. This is a concept not explicitly known to MINSTREL before its invention in The Vengeful Princess, and illustrates how MINSTREL’s creativity can extend its knowledge.

2.12 Issues in Creativity

This chapter has no doubt raised in the reader’s mind a number of issues about creativity and MINSTREL’s model of the creative process. This section addresses the more common questions about MINSTREL’s creativity model.

2.12.1 Errors in Creativity

The previous section showed how MINSTREL used creativity to invent a scene in which Jennifer wants revenge on Darlene because Darlene has some berries that Jennifer wants. Readers of this scene sometimes complain that possession of berries is insufficient motive for revenge. MINSTREL’s creative problem solving has invented an incorrect solution, at least according to some readers. How did this happen, and what does it say about the creative process?

When MINSTREL begins telling The Lady’s Revenge, it has only minimal knowledge about revenge. MINSTREL knows that revenge can be motivated by a thwarted goal, but it does not have any specific examples of revenge in the King Arthur domain in memory, and neither does it have any examples of thwarted goals in the King Arthur domain. Creativity must be used to extend MINSTREL’s knowledge about both thwarted goals and revenge.

MINSTREL’s creativity heuristics correctly extend its knowledge about possession of objects and thwarted goals. The scene in which Darlene takes the berries that Jennifer wants was created by TRAM:Opposite-State-Achieves from a scene that contained no thwarted goals. Initially MINSTREL knows that possessing an object can achieve a person’s goal of controlling an object. After telling The Lady’s Revenge, MINSTREL has discovered that possessing an object can also thwart another person’s goal of controlling an object.

The same creativity heuristic is less successful in extending MINSTREL’s knowledge about revenge. The thwarted possession of berries is invented as motivation for a revenge goal, but as most people recognize, that is probably insufficient motivation. Revenge is only plausible if the retribution is commensurate with the offense, or if the character seeking revenge is evil and likely to seek revenge out of proportion with the offense.
Chapter 2

Creativity extends a problem solver's knowledge. By making good use of what he already knows, a problem solver can often make accurate guesses about what he doesn't know. But sometimes he'll be wrong. That is the nature of creativity. Unlike first-order predicate logic and standard problem solving, creativity is a heuristic activity that trades infallibility for power. Creativity is able to discover many solutions that standard problem solving cannot because it takes risks. In this case, MINSTREL makes a reasonable guess about how the world operates—that possession of an object prevents possession by another, and that thwarting a possession goal is reason for revenge—but the guess is not completely correct.

Of course, this particular error could be easily corrected. MINSTREL could be given additional knowledge about revenge or possession of objects, and hence avoid this error. As this points out, creativity alone cannot extend a problem solver's knowledge indefinitely or without error. The problem solver must also be able to learn directly from outside experience. In a complete cognitive model, creativity must act in concert with noncreative learning to correctly extend a problem solver's knowledge of the world.

But no matter how much knowledge a problem solver has, creativity will extend beyond the borders of that knowledge, and errors of this type will continue to occur. The challenge is to find a model of creativity that will limit errors while still providing the power to discover novel, useful ideas.

MINSTREL's model of creativity limits errors by using creativity heuristics that make specific, small problem transformations. By limiting the changes made to a problem specification, MINSTREL increases the likelihood that any solution it discovers will apply correctly to the original problem. The success of this strategy is apparent even in MINSTREL's failures. "Possessing berries" may be a poor reason for revenge, but not an unreasonable one. The reader may judge it insufficient motivation, but he does at least understand the reasoning.

It is worth noting that in a complete cognitive system, errors in creativity would lead to learning experiences. If MINSTREL were able to learn from criticism, the reader's comments about the possession of berries being insufficient motivation for revenge would be an opportunity for MINSTREL to refine its knowledge about revenge and possession of objects, and avoid similar mistakes in the future. Creativity can thus be seen as a motivator and director of noncreative learning. At this time, however, MINSTREL has no ability to learn from criticism.

2.12.2 Learning in MINSTREL

Although MINSTREL does not have the ability to learn from criticism, it is able to do a simple sort of learning from creativity. When MINSTREL invents a new solution to a problem, the solution is indexed in episodic memory according to the problem it solves. So when MINSTREL discovers that a conflict over possession of berries is a reason for revenge, that new knowledge is stored in episodic memory, where it can be used in future problem solving. This serves several purposes.

First, it permanently extends MINSTREL's knowledge. If MINSTREL did not remember its past inventions, it would always begin its creative problem solving from the same base of knowledge, and would always explore the same area around the edges of its knowledge. But if a useful new solution is discovered and incorporated into MINSTREL's knowledge base, the new solution becomes an island from which MINSTREL can continue its creative exploration. By saving its creative successes, MINSTREL increases its ability to discover new solutions. The larger MINSTREL's episodic memory, the vaster its experience, the more powerful and effective its creativity.

Second, saving successful new solutions improves MINSTREL's efficiency as a problem solver. Standard case-based problem solving is very efficient, because it finds solutions that apply immediately and with a minimum of effort. Creativity is less efficient, because it must search the problem space and apply solution adaptations. By remembering past solutions, MINSTREL avoids having to reinvent them, and this improves MINSTREL's efficiency.

Both these benefits require MINSTREL to save successful solutions. Saving solutions that are wrong or even doubtful (like the "possession of berries" solution) can be counterproductive, because it will lead to MINSTREL repeating its past mistakes. It should be obvious, then, that criticism is an important element to a creative problem solver that learns from its creativity. Although MINSTREL does not currently address this issue, it is an area for future research.

Experiments that study MINSTREL's ability to learn from creativity and the effect that has on MINSTREL's problem solving behavior are presented in Chapter 8.

2.12.3 MINSTREL's Efficiency

Saving invented solutions improves MINSTREL's efficiency in the long run because it allows standard problem solving to solve problems that would otherwise require creativity. But how can we characterize the efficiency of creativity in the short run, that is, for a particular problem solving effort?

To begin with, creativity does not become inefficient as episodic memory grows. Retrieval from episodic memory is proportional to the number of significant features in the problem description, not upon the number of episodes in
memory. Episodic memory is organized as a tree based on the values of significant features of the indexed episodes (Kolodner, 1984; Reiser, 1986; Schank, 1982). Retrieval involves comparing the significant features of the recall description with the branches of this tree (i.e., traversing a multibranched tree). Since there is one comparison for each feature of the recall description, the time efficiency of recall is characterized by the number of features.

In fact, creativity tends to become more efficient as episodic memory grows. As memory grows, the likelihood that a transformed problem description will recall a solution increases, and so the likelihood of creativity succeeding also increases. There is no easy way to characterize this trend, because creativity does not search memory in an orderly fashion, and memory does not grow in an orderly fashion. But in general, the more the knowledge captured in episodic memory, the more likely it is that creativity will discover a solution. Some studies of how MINSTREL’s behavior changes as episodic memory changes are discussed in Chapter 8.

The efficiency of creativity depends primarily upon the number of creativity heuristics that are applicable to a particular problem. If a creative problem solver has three heuristics that apply to a problem, and tries every combination of these heuristics without finding a solution, the problem solver will try 24 different combinations. In the worst case (when no solution is found), creativity is \( O(\text{n}^3) \), where \( n \) is the number of applicable heuristics.

In practice, a creative problem solver is likely to limit the amount of time and effort expended to find a solution. MINSTREL, for example, applies no more than three heuristics simultaneously, regardless of how many are applicable. Because combinations of heuristics search the problem space farther and farther from the original problem specification, they are correspondingly less likely to discover a solution (although if they do it is likely to be quite different from known solutions). Consequently, the problem solver soon reaches a point of diminishing return and abandons the search for a solution.

Experience with MINSTREL also indicates that because MINSTREL’s creativity heuristics are very specific only a few of the available heuristics apply to any given problem. An analysis of the problems solved by MINSTREL in telling stories based on four different story themes revealed that of MINSTREL’s 24 creativity heuristics, on average only 1.4 TRAMs applied to any particular problem. This indicates that MINSTREL’s strategy of specific problem transformations not only increases the likelihood of discovering a useful solution, it also limits the effort expended in searching for a solution. (For experiments and studies on MINSTREL’s usage of TRAMs, see Chapter 8).

---

6 We assume a constant-time hashing function to traverse the memory tree.

2.12.4 Randomness in MINSTREL

A common question when people first hear about MINSTREL is “How does it make up the stories? Does it just make random choices?” As should be apparent to the reader of this chapter, MINSTREL solves problems (including the problem of telling a story) by purposeful, directed use of the knowledge in episodic memory. MINSTREL does not, in general, make random selections when problem solving or being creative.

The only use of random selection in MINSTREL occurs when MINSTREL must select between equally likely alternatives. When MINSTREL has no way to distinguish two or more choices, it selects randomly between the choices. This can occur in two situations.

First, episodic memory may recall two (or more) episodes which fulfill the recall criteria. Episodic memory is organized by the significant features of episodes. Two episodes with identical significant features will therefore fall into the same category in memory, and be recalled together. (If more than a small number of episodes fall into the same category, episodic memory builds and returns a generalization based on those episodes.) When this occurs, MINSTREL selects and uses one of the episodes randomly.

Second, MINSTREL selects randomly from its available plans and creativity heuristics when there is more than one applicable plan or heuristic. For any particular author-level goal there may be several author-level plans available. Similarly, for any particular problem solving situation, there may be several applicable creativity heuristics. In these situations, MINSTREL selects the plan or heuristic to apply randomly.

It is important to note that while MINSTREL does select randomly in these situations, MINSTREL does not make random decisions. When MINSTREL selects between two recalled episodes, it has already decided what it needs, and both episodes will fulfill those needs. Similarly, when MINSTREL selects an author-level plan randomly from those that apply to an author-level goal, it has already decided the goal. So MINSTREL’s random selections are not random decisions, merely a way to distinguish otherwise indistinguishable outcomes.