HOW COULD A COPYCAT EVER BE CREATIVE?

DOUGLAS HOFSTADTER
Center for Research on Concepts and Cognition
Indiana University Bloomington Indiana 47408 USA

Abstract. Analogy-making in a microdomain is used as the testbed for a computer model of creativity. Among the unusual features of the model are: (1) its nondeterministic parallel-processing architecture, modeled to some extent on architectures for perception; (2) the modeling of concepts as probabilistically overlapping regions; and (3) the fact that the system manufactures its own mutable representations of situations, rather than being fed fixed human-made representations. Statistical summaries of large numbers of runs on a few related problems reveal some of the program's "personality" and "esthetic taste". The program's response to a subtle analogy problem whose most insightful solution requires a radical perceptual leap tantamount to a miniature conceptual revolution is used to draw some general morals about the principles underlying successful paradigm shifts. Finally, the indispensability of calculated risk-taking for creativity is argued, and the intimate relationship between biased randomness and risk-taking is discussed.

1. On the Banality of Copycats

Can there be any doubt that to call someone a "copycat" is a taunt? Surely, copycats do nothing more than mechanical mimicry. Surely, a copycat could never be creative. After all, how could merely *doing the same thing* as someone else ever be original?

Consider the act of translating a novel from French to English. Readers of the translation — people who don't know a word of French — will proudly say, "Oh, yeah — I've read lots of Proust", as if the switch of medium were a triviality not worth mentioning. Some will even gush over his elegant use of language! After all, but for the words, isn't the English version the same as the French? More sophisticated is the remark, "I once read the entire English version of *A la recherche du temps perdu*". Yet even saying "*the* English version" still diminishes the act of translation, for it implies there is just one correct way of rendering the French into English. Many publishers the world over seem to endorse this view, either by not mentioning a book's translator at all, or by listing the translator's name only in fine print on the copyright page.

Is someone who deserves mention only on the copyright page no more than a copycat? Or are there cases where being a copycat is a significant achievement? Surely, the reconstruction of Proust's beautifully balanced but byzantine prose in a completely different linguistic and cultural medium is itself an act of marvelous precision requiring great skill. Surely, doing "what Proust did", but doing it in English, is a magnificent creative act.

Examples like this show that "doing the same thing" is a matter of perspective. Two things that are the same on a very abstract level may be wildly different, even incomparable, on other levels. Such mixtures of abstract sameness with concrete differentness are what analogy-making is all about. It can even be argued that this ability lies at the heart of all insightful thought (Hofstadter 1985, Chap. 24). Thus being a copycat and being creative can actually be very close. In recognition of this under-appreciated fact, the nondeterministic analogy-making computer program developed over a several-year period by Melanie Mitchell and myself is named "Copycat".

2. A Straightforward Copycat Problem

Copycat is intended to simulate human analogy-making and the mechanisms underlying it. Copycat operates in a carefully designed alphabetic microworld intended to bring out the central features of analogy-making while having very few domain-specific idiosyncrasies. That is, the reduction of analogy-making to the Copycat domain is intended to highlight the cognitive issues while deemphasizing noncognitive detail.

Copycat is not just intended to be a model of analogy-making (in itself a rather daunting task); it is intended to be a model of *creative* analogy-making. Indeed, our ambitions in the Copycat project and related projects (see Mitchell, 1993; Hofstadter, 1983; French, 1992; McGraw & Hofstadter 1993) go even further: to reach an understanding of the fundamental mechanisms of creativity in all types of cognition. To be sure, this is not a goal we will reach; however, it is surprising how deeply one can go into the problem in the tiny microworlds that these projects involve.

Before describing the architecture of Copycat, I exhibit the domain and show how the program actually performs on a couple of problems. Here is one of the simplest and most mundane analogy problems in the domain: Suppose the letter-string **abc** were changed to **abd**; how would you change the letter-string **ijk** in "the same way"? More concisely:

Problem 1. **abc** ⇒ **abd**
ijk ⇒ ?

Most people answer **ijl** (seeing the initial change as a replacement of the rightmost letter by its alphabetic successor), and an occasional "smart aleck" will answer **ijd** (pretending to see the change as a replacement of the rightmost letter by **d**), but it is rare for anyone to come up with anything else, though it has happened. (In theory, one could answer **abd**, but no one has ever actually done so except in trying to come up with a list, such

as this one, of many diverse potential answers.)

Since Copycat is nondeterministic, it follows different pathways on different runs, and thus comes up with a variety of answers. Figure 1 summarizes, in a bar graph, 1,000 runs. Each bar's height represents the frequency of the answer that is written beneath it; printed above the bar is the actual number of times the answer was produced.

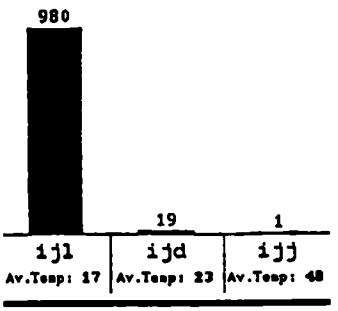

Figure 1. Results of 1,000 runs on Problem 1.

An answer's frequency indicates how *easily available* — that is, how *obvious* — the answer is to the program. For example, **ijl**, produced 980 times, is far more obvious to Copycat than **ijd**, produced just 19 times, which is in turn much more obvious than the strange answer **ijj**, produced only once. (To get this answer, the program replaced the rightmost letter by its predecessor rather than its successor.)

The nondeterministic decisions Copycat makes are all on a microscopic scale, compared with the large-scale decision of which answer to give. Although all runs are different, statistics lead to a quite deterministic behavior at the macroscopic level, as the bar graph shows: 98 percent of all runs on Problem 1 wind up in a single answer — **ijl**.

Getting one answer nearly all the time is not the only possible type of macroscopic determinism. Imagine a problem with two strong rival answers. From run to run, the program might be very unpredictable, oscillating erratically between the two answers, yet statistics collected over 1,000 runs might show the program chose one of the answers 60 percent and the other one 40 percent of the time. This *frequency pattern* would be a reproducible fact, in the sense that in another set of 1,000 runs, nearly the same percentages would re-arise.

The phenomenon of nearly-deterministic macroscopic behavior emerging from microscopic nondeterminism is often demonstrated in science museums by means of a contraption in which several thousand small balls are allowed to tumble down, one by one, through a dense grid of pins between two vertical plexiglass sheets. Each ball bounces helter-skelter downwards, eventually winding up in one of some 20 or 30 adjacent equal-sized bins forming a horizontal row at the bottom. As the number of balls that have fallen increases, the stack of balls in each bin grows. However, not all bins are equally likely destinations, so different stacks grow at different rates. In fact, the heights of the stacks in the bins at the bottom gradually come to form an excellent approximation to a perfect gaussian curve, with most of the balls falling into the central bins, and very few into the edge bins. This reliable build-up of the mathematically precise gaussian curve out of many unpredictable, random events is quite beautiful to watch.

In Copycat, the set of bins corresponds to the set of different possible answers to a problem, and the precise pathway an individual ball follows, probabilistically bouncing left and right many times before "choosing" a bin at the bottom, corresponds to the many stochastic micro-decisions made by Copycat during a single run. Given enough runs, a reliably repeatable pattern of answer frequencies will emerge, just as a near-perfect gaussian curve regularly emerges in the bins of a "gaussian pinball machine".

Although the frequencies in Figure 1 seem quite reasonable, it is not intended that they should precisely reproduce the frequencies one would find if this problem were posed to *humans*, since the program is not meant to model all the domain-specific mechanisms people use in solving these letter-string problems. Rather, what is interesting here is that the program exhibits plausible behavior, in that it nearly always comes up with answers that people find sensible, despite the fact that it has the *potential* to arrive at very strange answers (not just **ijj**, but many others). The seeming weakness of being able to produce large numbers of very odd answers is in fact a hidden strength of the Copycat architecture. To explain this is an important goal of this paper.

Printed below each answer is a number called the *average final temperature*, whose precise meaning will be explained later. For now, it can be thought of as an indication of the program's sense of the answer's *depth* or *quality*. The lower the final temperature, the deeper the program considers the answer to be. Copycat found **ijl**, with a final temperature of 17, to be the deepest answer; **ijd**, with 23, to be next-deepest; and **ijj**, with 48, to be far shallower. In this problem, then, Copycat's rank-ordering of answers by obviousness and by depth happen to agree.

3. A Challenging Copycat Problem

In the following problem, Copycat's rank-orderings by obviousness and by depth turn out not to agree.

Problem 2. **abc ⇒ abd**
xyz ⇒ ?

The focus here is clearly the **z**, the challenge being its lack of successor. Many people, eager to *construct* a successor to **z**, invoke the notion of circularity; thus they suggest **a** as the successor of **z**, much as January can be considered the successor of December, or '0' the successor of '9'. This gives **xya**.

Invoking circularity to deal with Problem 2 is undeniably a type of creative leap, and not trivial. However, the everyday notion of circularity is not available to the program, as it is to people, for borrowing and insertion into the alphabet world. In fact, it is stipulated that Copycat's alphabet is

linear and stops dead at z. We deliberately set this roadblock, since one of our main goals was to model the process whereby people deal with impasses.

People, in response to this "z-snag", come up with a wide variety of ideas, some creative and some not so, including these: replace the z by nothing at all, yielding answer xy. Or replace the z by the literal letter d, yielding xyd. (In other circumstances, such a resort to literality would be considered crude, but here it appears reasonable.) Other answers, too, are possible, such as xyz itself (since the z cannot move farther along, just leave it alone); xyy (since you can't take the *successor* of the z, why not take second best — its *predecessor*?); xzz (since you can't take the successor of the z itself, why not take the successor of the letter sitting next to it?); and numerous others.

There is one special way of looking at this problem that, to most people, seems like a deep insight, whether or not they themselves come up with it. This is the idea that abc and xyz are "mirror images" of each other, each one being "wedged" against its own end of the alphabet. That is, whereas one should read abc in the normal way (left-to-right), one should "flip" the reading direction of xyz to right-to-left. In other words, one would read it aloud as "z-y-x" rather than as "x-y-z". Such a flip turns xyz from being an *alphabetic* string into an *antialphabetic* one, a change in perspective that involves swapping not just the concepts *left* and *right* but also *successor* and *predecessor*. Such abstract swaps are called, in our model, *conceptual slippages*.

To anyone who had come so far as to see this flip, sticking with the rule "Replace the rightmost letter by its successor " and trying to use it on xyz would seem extraordinarily literal-minded. A more flexible approach would have the rule itself take into account the two conceptual slippages (*rightmost* ⇒ *leftmost* and *successor* ⇒ *predecessor*) involved in the reversal of xyz. This would imply the following "translation" operation on the rule:

"Replace the rightmost letter by its successor" ⇒
"Replace the leftmost letter by its predecessor"

The "translated rule", on the second line, leads to the answer wyz.

In my experience (confirmed by some limited experiments), most people consider this answer to be very elegant, and rank it higher than all the other answers proposed above. More than any other answer, it seems to be *doing the same thing* to xyz as was done to abc. Moreover, I claim that *creative* people tend to find this answer much more often than non-creative people, and even if they don't find it themselves, they appreciate it more than non-creative people do. To be sure, this is a subjective and perhaps even controversial statement, since there is no objective determiner or formal definition of creativity; however, if one is going to theorize and write about creativity, one must believe creativity is an objective phenomenon, and one must have at least a modicum of self-confidence that one can recognize it even in the absence of an objective measure or formal definition. Thus I stick my neck out and stand by my claim, stated in a new way:

The answer wyz to Problem 2 is a *creative* answer.

I will also make a second fairly bold claim: Not only does the act of coming up with wyz constitute a miniature case of the notion of "conceptual revolution" or "paradigm shift" (Kuhn, 1970), but it contains the *essence* of the phenomenon in a domain-independent manner. If this is true, then careful study of the process whereby humans (or machines) arrive at wyz could prove to be of importance in the study of creativity. Let us thus look at the origins of the perceptual flip of xyz.

It would seem that the flip must originate in the observation that the z's counterpart in abc is not the c but the a. But where would this flash of insight come from? Part of the answer is of course that a and z play intrinsically symmetric roles in the alphabet, but that cannot be the whole story. To see why, consider this very close cousin to Problem 2:

Problem 3. abc ⇒ abb
xyz ⇒ ?

Here, the rule "Replace the rightmost letter by its predecessor" not only describes what happened to abc elegantly and abstractly, but also applies effortlessly to xyz, yielding xyy. There is no z-snag and thus no pressure to see anything but the a–x and c–z connections — *bridges*, as we call them. This shows that the pressure to "flip" in Problem 2 must come in large part from the snag itself. (This is not to say that the answer yyz, which would involve the same pair of perceptual flips as in Problem 2, would be ugly or unacceptable here — in fact, it might even be more elegant than xyy — but it would seem unnecessarily clever, given that one can make a perfectly fine answer without going to that trouble.)

How exactly does the combination of the z-snag with the intrinsic symmetry of a and z suggest the perceptual reversal of xyz? Why do more creative people tend to see this reversal more easily? Why is this answer more esthetically pleasing than all the other answers? To answer these questions definitively would, I believe, bring one very close to the crux of an understanding of creativity.

4. Is Copycat a Creative Individual?

We now show Copycat's performance on Problem 2.

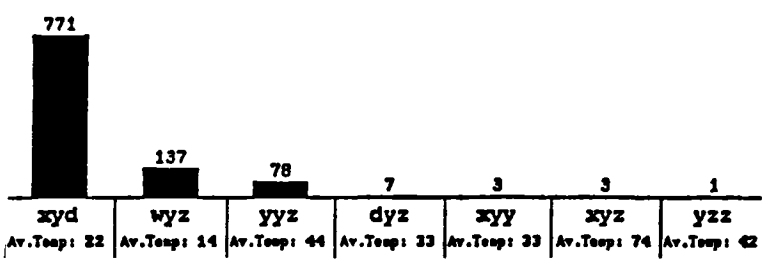

Figure 2. Results of 1,000 runs on Problem 2.

As can be seen, by far the most common answer given by Copycat is xyd, for which the program

decides that if it can't replace the rightmost letter by its *successor*, the next best thing is to replace it by a **d**. This is also an answer that people frequently give when told the **xya** avenue is barred.

A distant second in frequency, but the answer with the lowest average final temperature, is **wyz**. This discrepancy between rank-order by obviousness and rank-order by depth is characteristic of problems where creative insight is needed. Clearly, brilliance will distinguish itself from mediocrity only in problems where deep answers are elusive.

To bring about pressures that get the idea of the double reversal to "bubble up", radical measures must be taken upon hitting the **z**-snag. These include sharply focusing attention upon the trouble spot, and opening up a far wider range of possible avenues of exploration. Only with such a combination could **wyz** ever be found. The way this combination is evoked in Copycat is described in Section 7.

The next answer, **yyz**, reflects a view that sees the two strings as mapping to each other in a crosswise fashion, but ignores their opposite alphabetic fabrics; thus, while it considers the *leftmost* letter as the proper item in **xyz** to be changed, it clings to the notion of replacing it by its *successor*. Although this view seems somewhat inconsistent, like a good idea only carried out halfway, people very often come up with it. Indeed, such blends and half-completed thoughts are very characteristic of human cognition (Hofstadter & Moser, 1989).

The other four answers are much farther out on the low-frequency fringes. The answer **dyz** (which, appropriately enough, could be pronounced "dizzy"!) is a truly implausible blend of insight and simple-mindedness, the insight being the perception of the abstract symmetry linking **abc** and **xyz**, and the simple-mindedness being the idea that the **abc** ⇒ **abd** change is better seen in terms of the literal letter **d** than in the abstract terms of successorship.

In (Hofstadter & Gabora, 1990), a set of Copycat analogies (including this dizzy one) is used to illustrate a sketchy theory of "slippage humor". One tenet of the theory is that there is a continuum running from sensible analogies through "sloppy" answers and finishing in "dizzy" answers, where both "sloppy" and "dizzy" can be given semi-precise definitions in terms of how consistently conceptual slippages are carried out. Certain dizzy analogies, even in the austere Copycat domain, can actually make people laugh and can be convincingly mapped onto a class of real-world jokes. Any attempt to map out the multiple and complex interrelationships among insightful answers, reasonable answers, weak answers, far-out answers, erroneous answers, confused answers, and deliberately funny answers to analogy problems would be sure to bear rich fruit, even in this limited domain.

The answer **xyy** allows that the two strings are to be perceived in opposite *alphabetic* directions, yet refuses to give up the idea that they have the same *spatial* direction; it thus insists on changing the *rightmost* letter. It is amusing to note that **ijj** — Problem 1's analogue to this answer[1] — was given once in 1,000 runs, without any pressure of a snag.

The answer **xyz**, whose very high temperature of 74 indicates that the program considers it extremely shallow, comes from conceiving the initial change as "Replace **c** by **d**". (This is not nearly as clever as proposing that the letter **z** might serve as its *own* successor, which is an entirely different way of justifying this same answer, and one that people often suggest. In fact, when **xya** is barred, **xyz** is what people most often come up with.) Note that **ijk**, the analogous answer[1] to Problem 1, was *never* produced. It takes the "desperation" caused by the **z**-snag to allow such strange ideas any chance at all.

Finally, **yzz** is a peculiar, almost pathological, variant of the above-discussed answer **yyz**, in which the **x** and **y** in **xyz** are grouped together as one object, which is then replaced *as a whole* by its "successor" (the successor of each letter in the group). Luckily, it was produced only once in 1,000 runs, and was judged a poor answer by the program itself.

The existence of these rather zany fringe answers might be taken as a sign of weaknesses in the program's architecture, or at the very least as a sign of suboptimally-tuned parameters in the program. While this undoubtedly has some truth to it, I would argue that the occasional production of zany answers is a sign of Copycat's good "mental health".

To see why, one must understand that the Copycat architecture, described below, does not explore a single idea at a time; rather, it allows many types of exploratory mechanisms to "sniff" in parallel, with its perceptual discoveries dynamically regulating their relative speeds (*i.e.*, levels of involvement). A problem's fringe answers thus provide glimpses of the mechanisms that have been nearly, but not totally, suppressed. (It is of course crucial that mechanisms suppressed in context A be able to be evoked and even to play a dominant role in context B, and vice versa.) To *totally* suppress fringe answers would require such absolute quashing of rival mechanisms that the parallel architecture would be turned into a single-minded serial architecture, completely defeating its purpose.

Evidence for this type of parallelism in the human mind is provided by the important psychological phenomena of *contamination* and *blending* (Hofstadter & Moser, 1989). Any cognitive model that did not exhibit these phenomena at all would be suspect.

It is tricky to get appropriate mechanisms to "wake up" and dominate the processing in one context, yet to remain mostly dormant in nearby contexts. This context-sensitivity of mechanism evocation is typified by Copycat's performance on the next problem.

[1] It is ironic that a claim of analogousness of answers to different Copycat problems (such as the casual claim made in the text that **ijj** in Problem 1 is "the analogue" of **xyy** in Problem 2) comes across as objective and unproblematic to most readers, despite the fact that the mystery of how people make and judge such letter-string analogies is precisely the cognitive issue under discussion. Why do readers not object to such a claim and say, "*Who says* this answer is the analogue of that answer?"?

5. Effects of the Variation of Pressures

In Problem 2, pressure for a "crosswise" mapping comes both from the existence of an impasse and from the existence of an appealing way out of it — the intrinsic symmetry of **a** and **z**. Suppose the impasse was retained but the appeal of the "escape route" was reduced — what would be the effect? The following variant explores that question.

Problem 4. **rst ⇒ rsu**
xyz ⇒ ?

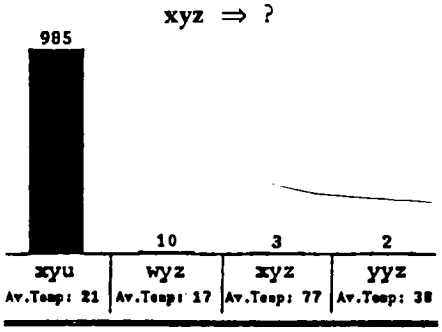

Figure 3. Results of 1,000 runs on Problem 4.

As Figure 3 shows, **wyz** was produced on only 1 percent of the runs, whereas in Problem 2 it was given on almost 14 percent of the runs. Here, there is very little to suggest building a crosswise bridge, because the **r** and **z** have almost nothing in common, aside from the fact that the **r** is leftmost in its string and the **z** rightmost in *its* string. But that is hardly a powerful reason to map the **r** onto the **z**.

For perspective, compare this to Problem 1. How much appeal is there to the idea of mapping the leftmost letter of **abc** onto the rightmost letter of **ijk**? Such a crosswise **a–k** bridge would result in either **hjk** or **jjk** (or possibly **djk**). But in 1,000 runs, Copycat never produced any of those answers, nor have we ever run into a human who gave any of them as an answer to Problem 1. A crosswise mapping in Problem 1 is completely "off the wall".

Things are different in Problem 4 because there is a snag that triggers emergency measures, which make the normally unappealing **r–z** bridge a bit more tempting, and so, once in a while, it gets built. From there on, the paradigm shift goes through just as in Problem 2, resulting in **wyz**.

In a sense, Problem 4 is poised midway between Problems 1 and 2, so answering **wyz** in it represents an intermediate stage between unmotivated and motivated fluidity. It is gratifying that Copycat responds to the different sets of pressures in these problems in much the way intuition says it ought to.

6. A Sketch of Copycat's Architecture

To go more deeply into the paradigm shift, we must refer to the mechanisms that underlie it in our model. Thus we now turn to Copycat's architecture. (A much fuller account is found in Mitchell, 1993.)

6.1 The Slipnet and Emergent Concepts

The Slipnet is the site of the program's concepts; as such it is Copycat's long-term memory. It is a network whose nodes are connected by links with dynamically varying lengths. Links represent conceptual associations. Some of the main concepts are: **a**, **b**, ..., **z**, *letter, sameness, successor, predecessor, alphabetic-first, alphabetic-last, left, right, direction, leftmost, rightmost, middle, group, sameness-group, successor-group, predecessor-group, group-length, 1, 2, 3, opposite, alphabetic-position, letter-category*, and so on.

A concept, rather than being represented solely by one node, consists of a blurry region called an "associative halo", which is *centered* on a particular node. The halo of a given node does not have sharp edges; rather, it includes various nearby nodes probabilistically, depending on how close they are to the central node. (This is reminiscent of the quantum-mechanical "electron cloud" in an atom, whose probability density falls off with increasing distance from the nucleus.) The distance between two nodes is essentially the path length between them (the sum of link-lengths). The function of an associative halo is *to permit conceptual slippages.* The closer two concepts are in the Slipnet, the more likely it is that either of them can "slip" into the other, under appropriate pressures. The existence of a blurry zone where either of two concepts can serve as the other smears concepts out probabilistically; this is why we insist a concept is not a single node.

Slipnet nodes support dynamic degrees of *activation.* A node becomes activated when instances of it are perceived in the Workspace (by codelets, as described below), and loses activation unless its instances remain salient. A node spreads activation to nearby nodes as a function of their proximity. Activation is not an on-and-off affair, but varies continuously. However, when a node's activation crosses a certain critical threshold, the node has a probability of jumping discontinuously into a state of full activation, from which it proceeds to decay.

Each node in the Slipnet has a *conceptual depth* — a static number intended to capture the generality and abstractness of the concept. For example, the concept *opposite* is deeper than *successor,* which is deeper than the concept **a**. Deep concepts are those that people are prone to use in describing the *essence* of situations. Therefore, once a deep aspect of a situation is perceived, it should have more influence on further perception than other aspects of the situation. Accordingly, activation decay-rates are such that *deep concepts decay slowly, shallow concepts rapidly.* Once a concept has been perceived as relevant, then the deeper it is, the longer it will *remain* relevant, hence the more profound an influence it will exert on the developing view.

Assignment of conceptual depths amounts to an *a priori* ranking of "best-bet" concepts. The idea is that a deep concept (such as *opposite*) is hidden from the surface and cannot easily be brought into the perception of a situation, but that once it *is* perceived, it should be regarded as highly significant. There is of course no guarantee that concepts deemed deep will be relevant in any *particular* situation, but such concepts were assigned high depth-values precisely

because we saw that they tended to crop up over and over again across many diverse types of situations, and we also noticed that the best insights in many important problems could be traced to their use. We therefore built into the architecture a strong drive, if a deep aspect of a situation is perceived, to use it and to let it influence further perception of the situation.

Conceptual depth has another important aspect: the deeper a concept is, the more resistant it is (all other things being equal) to slipping into another concept. That is, the program has a propensity to slip *shallow* concepts rather than deep ones, when some slippage has to be made. This idea can be summarized in a motto: "Deep stuff doesn't slip in good analogies." There are, however, interesting situations in which unusual constellations of pressures cause this motto to be disrespected; in fact, Problem 2 is one such situation.

There are a variety of *link types,* and for each given type, all links of that type share the same *label,* a label being itself a concept in the network. Every link is continually adjusting its length according to the activation level of its label, with high activation giving rise to short links, low activation to long ones. Stated another way: If concepts A and B have a link of type L between them, then as concept L's relevance goes up (or down), concepts A and B become conceptually closer (or further apart). An example of a label is the node *opposite,* which labels the link between nodes *right* and *left,* the link between nodes *successor* and *predecessor,* and several other links. If *opposite* gets activated, all these links will shrink in concert, rendering the potential slippages they represent more probable.

Conceptual proximity in the Slipnet is thus context-dependent. For example, in Problem 1, no pressures arise that bring the nodes *successor* and *predecessor* into close proximity, so a slippage from one to the other is highly unlikely; by contrast, in Problem 2, there is a good chance pressures will activate the concept *opposite.* If so, the link between *successor* and *predecessor* will shrink, bringing each one more into the other's halo, and enhancing the probability of a slippage between them.

The perceived *relevance* of a given concept — its probabilistic involvement in guiding perceptual processing — is a real number: the node's current activation level. This means there is not a yes-or-no answer to the question "Is concept X relevant at the present moment?"; the fact that probabilities as well as activation levels are *continuous variables* allows concepts to have arbitrary *shades* of relevance. In any problem, all the system's concepts have the potential, *a priori,* to be brought in and used; which ones actually *do* become relevant and to what degree depends on the situation, and of course on the particular pathway that the processing follows.

6.2 Perceptual Processing in The W orkspace

To my knowledge, Copycat is unique among current analogy-making programs in that it builds up its own representations of situations, as opposed to being given frozen human-made representations of situations. Moreover, its representations remain fluid throughout a run. The architecture allowing this to happen is much like a perceptual architecture, only at a higher level of abstraction.

The locus of Copycat's perceptual activity is the Workspace. As such, it contains instances of various concepts from the Slipnet, combined into temporary perceptual structures (*e.g.,* raw letters, descriptions, bonds, groups, and bridges). It can be thought of as Copycat's short-term memory, and resembles the global "blackboard" data-structure of Hearsay II. As in Hearsay, perceptual structures in the Workspace are built up gradually and hierarchically on top of the raw input (the trio of given letter-strings).

Any structure has, at any time, both a *salience,* which determines its probability of being noticed and built into larger structures, and a *strength,* which determines its probability of winning a fight with a rival structure. Though it might seem crass, the architecture honors the old motto "The squeaky wheel gets the oil", even if only probabilistically so.

Salience is a function of an object's descriptions — the more descriptions it has, and the more activated the concepts involved therein, the more salient the object will be. An object's strength is a function of the depth of the concepts in its descriptions, as well as the degree to which it is integrated with other objects around it. An object that does not mesh well with its neighbors will tend to have a low strength.

All processing in the Workspace is performed by small agents designed to carry out specific micro-tasks, such as attaching descriptions to objects (*e.g.,* attaching the descriptor *middle* to the **b** in **abc**), bonding two objects together (*e.g.,* inserting a *successor* bond between the **b** and **c** in **abc**), making groups out of adjacent objects bonded together in a uniform manner, making bridges that join similar objects in distinct strings (similarity being measured by proximity of descriptors in the Slipnet), destroying groups or bonds, and so on. These micro-agents, dubbed *codelets,* are born into and sit on the Coderack, waiting to be called. A codelet is a short piece of code that carries out some small, local task that is part of the process of building a structure (*e.g.,* one codelet might notice that the two **r**'s in **mrrjjj** are instances of the same letter, and propose a *sameness* bond between them; another codelet might estimate how well that proposed bond fits in with already-existing bonds; yet another codelet might build the bond; a later series of codelets might chunk the two bonded **r**'s as a sameness-group).

Any codelet, when created, is assigned an *urgency* — a number that determines its likelihood of being selected *next codelet to be run.* The reason behind the perhaps puzzling-seeming decision to select codelets probabilistically rather than deterministically (in order of urgency, for instance) is subtle but crucial; I believe that it lies at the crux of mental fluidity. This idea will be discussed in detail in Section 8.

Any structure, no matter how simple, is built by a series of codelets running in turn, each one

deciding probabilistically on the basis of its estimation of the proposed structure's strength whether to continue, by generating one or more follow-up codelets, or to abandon the effort at that point. Depending on its urgency, a follow-up codelet will have either a short time or a long time to wait before it can run and continue the pathway towards building some particular structure.

This type of breakup of structure-building processes serves two purposes: (1) it allows many such processes to be carried out in parallel, by having their components interleaved; and (2) it allows the computational resources allocated to any process to be dynamically regulated by moment-to-moment estimates of the promise of the pathway being followed. This means that many pathways can be at least "sniffed" without being deeply explored, and any exploration that seems to be leading nowhere can be quickly decelerated, or even "snuffed".

It is critical to understand that a process is not a predetermined macroscopic act that is then broken up into convenient chunks; rather, any sequence of codelets that amounts to a coherent macroscopic act can *a posteriori* be labeled a process — thus processes are *emergent*. The speed of any such process emerges dynamically from the urgencies of its component codelets. The upshot is a *parallel terraced scan* — more-promising views tend to be explored faster than less-promising ones.

Codelets come in two types: *bottom-up* and *top-down*. A bottom-up codelet seeks regularities of any sort, without prompting from a specific concept; a top-down codelet, by contrast, seeks instances of a specific active concept, such as *successor* or *successor-group*. The probability that a node in the Slipnet will add a top-down codelet to the Coderack is a function of that node's current activation level.

Any run starts with a standard initial population of bottom-up codelets having preset urgencies; at each time step, one codelet is chosen to run and is taken off the Coderack. The choice is probabilistic, based on relative urgencies. As the run proceeds, new codelets are added to the Coderack either as follow-ups to previously-run codelets or as top-down scouts for active concepts. A new codelet's urgency is assigned by its creator as a function of the estimated promise of the task it is to work on. As the run proceeds, the codelet population shifts in response to the system's needs as judged by previously-run codelets and by activation patterns in the Slipnet, which themselves depend on what structures have been built. This means there is a *feedback loop* between perceptual activity and conceptual activity, with observations serving to activate concepts, and activated concepts in return biasing the directions in which perceptual processing tends to explore. There is no top-level executive directing the system's activity; all acts are carried out by ant-like codelets.

6.3 Temperature and Controlled Randomness

A final mechanism, *temperature*, controls the degree of randomness used in making decisions (*e.g.*, which codelet to run next, which object to focus attention on, which of two rival structures should win a fight, etc.). The higher the temperature is, the less respected are *a priori* probabilities, based on urgencies, saliences, and so forth; conversely, the lower the temperature is, the more such judgments are respected. In the theoretical limit of infinitely high temperature, all avenues become equiprobable; in the opposite limit of zero temperature, all decisions become deterministic, with a higher-urgency or higher-salience choice always beating out its rivals. In practice, however, neither of these temperature extremes is ever reached.

The temperature at any moment is a function of the amount and quality of structure built so far. That is, temperature measures the degree of perceptual organization in the system. Temperature is high when little order has been discovered, meaning there is little information on which to base decisions; it is low when much order has been discovered, meaning there is greater certainty about the basis for decisions. Note that by controlling its own degree of randomness, the system governs the degree to which it is willing to take risks.

It can now be seen why *final* temperature — the temperature at the end of a run — serves as a good measure of how well the program "likes" an answer it finds: if there is very little high-quality structure, the answer has little justification, and so is of low quality. Conversely, the existence of much good structure means the answer has a solid and systematic conceptual underpinning, hence is deep.

7. Anatomy of a Miniature Paradigm Shift

We are now in a position to describe how Copycat can, on occasion, come up with the answer **wyz** to Problem 2. It turns out to be a surprisingly intricate tale. Our tale is also an attempt to show in slow motion how a human mind, under severe pressure, can radically transform its perception of a situation in a blinding flash (this is often colloquially called the "Aha!" phenomenon). Since such paradigm shifts are often found at the core of creative acts, one should expect their microstructure to be very subtle (otherwise the mystery of creativity would long ago have been revealed and put on a mass-produced microchip!). Indeed, the challenge of getting Copycat to produce **wyz** *properly* — faithfully to what we believe really goes on in a human mind at the most informative subcognitive level of description — has, from the very outset, been the central inspiration in guiding the development of the Copycat architecture.

7.1 Emergency Measures Triggered by an Impasse

Things start out analogously to a typical run on Problem 1 in terms of bonding, grouping, bridge-building, and such — that is, both **abc** and **xyz** are quickly perceived as left-to-right successor groups, and the rule "Replace the rightmost letter by its successor" is effortlessly produced. Everything proceeds smoothly up to the point of trying to take

the successor of **z**, which is impossible. This snag causes several coordinated "emergency measures" to be taken:

- the *physical* trouble spot — here, the instance of the letter **z** in the Workspace — is highlighted, in the sense that its salience is suddenly pumped up so high that, to codelets, it becomes the most attractive object in the Workspace;
- the *conceptual* trouble spot — here, the node **z** in the Slipnet — is highlighted, in the sense that a huge jolt of activation is pumped into it, and as a consequence, its halo broadens and intensifies, meaning that related concepts are more likely to be considered, at least fleetingly;
- the temperature is pumped up to the very high value of 100 and temporarily clamped there, thus encouraging a much broader and more open-minded search than normal;
- the high temperature enables previously dormant "breaker" codelets to run, whose purpose is to arbitrarily break structures that they find in the Workspace, thus reducing the system's attachment to a viewpoint already established as being problematic.

Note the generality of these "impasse-handling" mechanisms: they have nothing to do with this snag itself, with the particular problem, with the alphabetic domain, or even with analogy-making! The reason for this is that running into an impasse is a common type of event that any cognitive system must be capable of dealing with. To be sure, no set of mechanisms can be guaranteed to resolve *all* snags (otherwise we would be dealing with omniscience, not intelligence). The best that can be hoped for is that the impasse can be "read" as a source of cues — possibly very subtle ones — that may launch tentative forays down promising new avenues. A "cue", in the Copycat architecture, is essentially the creation of a *pressure* that pushes for exploration along a certain direction. Thus the idea of *interpreting the snag as a source of pressures* is the philosophy behind the four mechanisms above, especially the first two.

Although these emergency measures are not powerful enough to guide Copycat to coming up with **wyz** all that often, when it *does* produce **wyz**, it does so essentially according to the scenario described in the next several subsections.

7.2 New Biases Cause New Concepts to Bubble up

The spotlight focused on the Slipnet node **z** has the effect of making all concepts in its halo — including the closely-related concept *alphabetic-last* — more likely to be paid attention to by description-building codelets. The probability thus rises that the instance of **z** in the Workspace will get explicitly described as *alphabetic-last.*

Note that in most problems — even ones that involve one or more **z**'s — there is little or no reason to pay attention to the notion *alphabetic-last,* so this conceptually deep neighbor-concept of **z** usually remains — and *should* remain — dormant. After all, it is crucial to avoid cluttering up the processing with all sorts of extraneous interfering notions, no matter how conceptually deep they may be. But now, under the emergency measures, unusual avenues are more likely to at least be "sniffed out" a short ways.

If the description *alphabetic-last* does indeed get attached to the **z**, which is a dicey matter, then a further boost of activation is given to the node *alphabetic-last,* because the system has deemed it potentially relevant. Now, with *alphabetic-last* (part of the halo of **z**) lifted out of dormancy, concepts in *its* halo will in turn receive more activation, which means codelets will tend to pay more attention to them (probabilistically speaking). One such neighbor-concept is *alphabetic-first,* which is now briefly given the chance to show its relevance. Obviously, if there were no instance of **a** in the problem, *alphabetic-first* would be found to be irrelevant and would swiftly decay back to dormancy; however, since there *is* an **a** inside **abc**, it has a fair chance of getting explicitly described as *alphabetic-first,* in much the same way as the **z** in **xyz** got described as *alphabetic-last.*

If both these descriptions get attached — a big "if" — then both letters become even more salient than before; in fact, they almost cry out to be mapped onto each other — not because the system can anticipate the great insight that such a mapping will bring, but simply because both letters are so salient! Once the system tries it out, however, the great appeal of the tentative mapping instantly becomes apparent. Specifically, a pair of conceptual slippages are entailed in the act of "equating" the **a** with the **z** (*i.e.*, building an **a**–**z** bridge): *alphabetic-first* ⇒ *alphabetic-last,* and *leftmost* ⇒ *rightmost.*

7.3 Overcoming Resistance to a Deep Slippage

Although the deep slippage of *alphabetic-first* into *alphabetic-last* would normally be valiantly resisted (recall the motto "Deep stuff doesn't slip in good analogies"), here a special circumstance renders it a bit more probable: the companion would-be slippage *leftmost* ⇒ *rightmost* is of the same type — in particular, each of these slippages involves slipping some concept into its *opposite.* These two would-be slippages are thus conceptually parallel, so that each one on its own reinforces the other's plausibility. This fact helps to overcome the usual resistance to a deep slippage. (Incidentally, this is the kind of subtlety that was not apparent to us before the computer implementation was largely in place; only at that point were Copycat's flailings and failures able to give us pointers as to what kinds of additional mechanisms were needed.)

Another fact that helps overcome the usual resistance to this deep slippage is that *any* two slippages, whether parallel or not, provide more justification for building a bridge than either one alone would. Altogether, then, there is a fairly good chance that this bridge, once tentatively

suggested, will actually get built. Once this critical step has taken place, it's all downhill.

7.4 Locking-in of a New View

The first thing that is likely to happen as a result of an **a–z** bridge getting built is that the temperature will get unclamped from its high value of 100. In general, what unclamps the temperature is the construction of any strong structure different from those that led up to the snag — in other words, a sign that an alternative way of looking at things may be emerging. The unclamping of temperature is probabilistic: the stronger the novel structure is, the more likely it is to trigger the unclamping.

Since the **a–z** bridge is both novel and very strong, unclamping is virtually assured, which means the temperature falls drastically right after the bridge is built. Recall that when temperature falls, decisions get more deterministic, which means that the emerging new view will tend to get supported. In short, there is a powerful *locking-in effect* triggered by the discovery of an **a–z** bridge.

Another aspect of locking-in is the following idea. The building of this first bridge involving the simultaneous slippage of two concepts into their opposites sends a burst of activation into the deep concept *opposite*; as a result, all pairs of concepts connected by links labeled *opposite* are drawn much closer together, facilitating the slippage of one into the other. Such slippages will still not happen without reason, of course, but now they will be much easier to make than in ordinary circumstances. Thus in a sense, making *one* bridge based on conceptual opposites sets a tone making it easier to make *more* of them. The emerging theme of *oppositeness* might fairly be characterized as a kind of "bandwagon".

Given all this, one of the most likely immediate consequences of the crosswise **a–z** bridge is the building of the "mirror" crosswise bridge connecting the **c** with the **x**. It, too, depends on the slippage between *leftmost* and *rightmost*, and is thus facilitated; in addition, once built, it strongly reinforces the growing bandwagon for *opposite*. Moreover, the temperature will fall significantly because this bridge, too, will be very strong. Thanks to all this, the locking-in effect may be so strong that it will be hard to stop the momentum toward building a completely new view of the situation.

The reversals taking place become a near-stampede at this point, with significant pressure emerging to flip the direction of the fabric of the group **xyz** from *rightwards* to *leftwards*, which means also that the perceived fabric itself will switch from *successor* to *predecessor*. Thus at this point, Copycat has carried out both a spatial and an alphabetical reversal of its vision of **xyz**. The paradigm shift has been completed. Copycat is now ready to translate the raw rule, and, as was said above, the result is the new rule *replace the leftmost letter by its alphabetic predecessor*, which yields the answer **wyz**.

It must be stressed that all the multifarious activity just described — shifting levels of activation of various key concepts; deep slippages; interrelated spatial and conceptual reversals — all this takes place in a flash in a human mind. There is no hope of making out all the details of this paradigm shift (or any other) in one's own mind through mere introspection. In fact, it has taken us several years to settle on the above account, which represents our current best stab at the true story's essence. (For another close-up view of all this, see the annotated series of screen dumps of a particular run on Problem 2 given in Mitchell, 1993.)

7.5 The Ironic Strength of Deep Slippages

As was pointed out a moment ago, the motto "Deep stuff doesn't slip in good analogies" is violated by the answer **wyz**, in that *alphabetic-first* is a deep concept and yet is allowed to slip into *alphabetic-last* here. This, I speculate, is one reason that makes it so hard for many people to discover it on their own. Yet many people, when they are shown this answer, appreciate its elegance and find it very satisfying. Problem 2 is thus a circumstance where a constellation of pressures can (at least occasionally) overcome the powerful natural resistance expressed by the motto; in fact, making such a daring move results in what many people consider to be a deep and insightful analogy.

There is an important irony here. Even though slippages tend to be (and should be) *resisted* in proportion to their depth, once a very deep slippage has been made, then it tends to be (and should be) *respected* in proportion to its depth. We consider this to be characteristic of creative breakthroughs in general. Indeed, we consider the process of arriving at answer **wyz** to be "the same", just in miniature, as the process whereby a full-scale conceptual revolution takes place in science.

8. Randomness in Service of Intelligence

Many people feel very uneasy with the proposition that greater intelligence can result from making *random* decisions than from making *systematic* ones. Indeed, when Copycat's architecture is described this way, it sounds nonsensical. Isn't it always wiser to choose the *better* action than to choose *at random*? However, as in so many discussions about minds and their mechanisms, this appearance of nonsensicality is an illusion caused by a confusion of levels.

Certainly it would seem counterintuitive — in fact, downright nonsensical — if someone suggested that a melody-composition program (say) should choose its next note by throwing dice, even weighted dice. How could any global coherence come from such a process? This objection is of course totally valid — good melodies cannot be produced in that way (except in the absurd sense of millions of monkeys plunking away on piano keyboards and, once in a blue moon, coming up with "Blue Moon"). But the Copycat architecture in no way advocates such a baldly random type of decision-making procedure.

The choice of next note in a melody is a *top-level* decision, as opposed to a low-level act of "micro-exploration". The purpose of micro-exploration is to efficiently explore the vast, foggy world of possibilities lying ahead without getting bogged down in a combinatorial explosion; for this purpose, randomness, being equivalent to non-biasedness, is the *most efficient* method. Once the terrain has been scouted out, then a lot of information has been gained, and in most cases some macroscopic pathways have been found to be much more promising than others. Moreover — and this is critical — the more information that has been uncovered, the more the temperature will have dropped — and the lower the temperature is, the less randomness is used. In other words, the more confidently the system believes, thanks to lots of efficient and fair micro-scouting in the fog, that it has identified a particular promising pathway ahead, the more certain it is to make the macro-decision of picking that pathway. Only when there is a very tight competition is there much chance that the favorite will not win, and in such a case, it hardly matters, since even after careful exploration, the system is not persuaded that there is a clear best route to follow.

In short, in the Copycat architecture, hordes of random forays are employed on a microscopic level when there is a lot of fog ahead, and their purpose is to get an evenly-distributed sense of what lies out there in the fog rather than simply plunging ahead blindly, at random. The foggier things are, the more unbiased should be the scouting mission, hence the more randomness is called for. To the extent that the scouting mission succeeds, the temperature will fall, which in turn means that the well-informed macroscopic decision about to be taken will be made *non*-randomly. Thus, randomness is used *in the service of*, and not in opposition to, intelligent nonrandom choice.

In a complex world (even one with the limited complexity of Copycat's microworld), one never knows in advance what concepts may turn out to be relevant in a given situation. It is therefore imperative not only to avoid dogmatically *open-minded* search strategies, which entertain all possibilities equally seriously, but also to avoid dogmatically *closed-minded* search strategies, which in an ironclad way rule out certain possibilities *a priori*. Copycat opts for a middle way, in which it quite literally takes calculated risks all the time — but the *degree* of risk-taking is carefully controlled. Of course, taking risks by definition opens the potential for disaster — and indeed, disaster occurs once in a while, as was evidenced by some of the far-fetched answers, like **ijj** in Problem 1 and **dyz** and **yzz** in Problem 2. But that is the price that must be paid for flexibility and the potential of creativity.

People, too, occasionally explore and even favor peculiar routes. Copycat, like people, has to have the potential to concoct strange and unlikely solutions, in order to be able to discover subtle and elegant ones, such as **wyz** in Problem 2. To rigidly close off any routes *a priori* would necessarily remove critical aspects of Copycat's flexibility. On the other hand, the fact that Copycat so rarely produces strange answers demonstrates that its mechanisms manage to strike an effective balance between open-mindedness and closed-mindedness, imbuing it with both flexibility and robustness.

Two experiments were done to determine the extent to which Copycat's ability to carry out paradigm shifts actually depends on temperature. In each experiment, temperature was clamped, first at the high value of 100, then at the low value of 10. The problem used in the experiments was this one:

Problem 5. **abc** ⇒ **abd**
mrrjjj ⇒ ?

One might think the deep answer here is **mrrkkk** (obviously far deeper than the literal-minded **mrrjjk** and **mrrjjd**), but by moving away from the surface level of *letters* to the more hidden level of *group-lengths*, one uncovers a "1-2-3" pattern, which allows a far crisper and richer mapping of **mrrjjj** onto **abc**. This radical perceptual shift leads to the answer **mrrjjjj**, which Copycat finds about 4 percent of the time, with a temperature far lower than for **mrrkkk**.

In each test, 200 runs were made. In neither test did the hobbled Copycat ever find **mrrjjjj** — not even once. The experiments thus vindicated the idea, central to Copycat's philosophy, that controlled risk-taking on a micro-level is essential for the making of creative breakthroughs on a macro-level.

Certainly a key aspect of creativity is the ability to pinpoint exactly where an invalid but tacit assumption has been made, and to "slip" that assumption in the proper way. How does one "unbury" just the right tacit assumption without a brute-force search through *all* assumptions? And how about the fact that not all tacit assumptions are available, not even to a brute-force search, since they may be encoded as *procedural* rather than declarative knowledge? It is to be hoped that the analysis of Copycat's performance on the tiny Problem 2 and its cousins will shed light on these questions that lie at the heart of genuine, full-fledged creativity.

References

French, R. M. (1992) *Tabletop: An Emergent Stochastic Computer Model of Analogy-making.* Ph.D. thesis, University of Michigan, Ann Arbor.

Hofstadter, D. R. (1983) "The architecture of Jumbo". In *Proceedings of the International Machine Learning Workshop.* Monticello, Illinois.

Hofstadter, D. R. (1985) *Metamagical Themas: Questing for the Essence of Mind and Pattern.* Basic Books, New York.

Hofstadter, D. R. and Moser, D. J. (1989) "To err is human; to study error-making is cognitive science". *Michigan Quarterly Review 28* (2), pp. 185–215.

Kuhn, T. S. (1970) *The Structure of Conceptual Revolutions* (second edition). U. of Chicago Press, Chicago.

McGraw, G. E. and Hofstadter, D. R. (1993) "Perception and creation of diverse alphabetic styles". In this volume.

Mitchell, M. (1993). *Analogy-Making as Perception.* Bradford Books/MIT Press (in press), Cambridge, Massachusetts.