Northwestern University

The Institute for the Learning Sciences

TOWARDS A GENERAL THEORY OF PLANNING AND DESIGN

Technical Report # 44 • September 1993

Lawrence Birnbaum
Gregg Collins

Established in 1989 with the support of Andersen Consulting

Towards a General Theory of Planning and Design

Lawrence Birnbaum and Gregg Collins

September 1993

The Institute for the Learning Sciences
Northwestern University
1890 Maple Avenue
Evanston, Illinois 60201

This work was supported in part by by the Air Force Office of Scientific Research under grant no. AFOSR-91-0341-DEF, and by the Advanced Research Projects Agency, monitored by the Office of Naval Research under grant no. N00014-91-J-4092. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of The Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech and North West Water Plc, Institute Partners.

Towards a General Theory of Planning and Design

Lawrence Birnbaum and Gregg Collins

Northwestern University
The Institute for the Learning Sciences
Evanston, Illinois

1 Introduction

The development of artificial intelligence depends upon progress in both the architecture of intelligent systems and representation of knowledge. It is therefore not surprising that the areas of AI that have advanced most dramatically in recent years are those in which progress on architecture has gone hand in hand with progress on representation. Thus, for example, qualitative physical reasoning has made impressive strides both in architecture, as exemplified by assumption-based truth maintenance systems (deKleer, 1986) and qualitative simulators (e.g., QPE [Forbus, 1990] and QSIM [Kuipers, 1986]), and in knowledge representation, as exemplified by Forbus's (1984) Qualitative Process theory and deKleer and Brown's (1985) qualitative physics based on confluences. The increasing competence of computer problem-solving systems in physical domains is directly attributable to this balance in effort and progress between architecture and representation.

In striking contrast, however, research in AI planning systems has exhibited no such healthy balance between architectural and representational concerns. On the contrary, the overwhelming majority of work in the field has been aimed almost exclusively at the design of architectures.[1] As a direct result, current AI planning systems can neither represent nor utilize general planning concepts such as "bottleneck," "milestone," or "crisis," commonly employed by human planners in reasoning about and discussing planning problems.

This inattention to the content and structure of general planning knowledge has had three unfortunate consequences. First, as has been widely noted, AI planning systems remain for the most part mired in simplistic domains such as blocks world and tileworld, for the simple reason that they are unable to represent anything more complex. Second, the work on architecture has itself been distorted by the lack of corresponding progress on the representational front, so that "general purpose" planning models have essentially been reduced to programming languages, embodying hardly any knowledge about planning,

[1] See, for example, Fikes and Nilsson (1971), Sacerdoti (1977), Chapman (1987), Wilkins (1988), and McAllester and Rosenblitt (1991).

and no knowledge about the world. As a result, these models have failed to provide the hoped-for leverage in adapting planners to new domains that motivated their development in the first place. And finally, because these systems are unable to reason in terms of commonplace planning abstractions, they can neither explain themselves to, nor take advice from, human planners. For these reasons, we believe that the critical issue in planning at this time is to place representation on an equal footing with architecture by articulating and taxonomizing the general planning and design knowledge of human planners.

1.1 What is general planning knowledge?

We believe that a central goal of planning research should be the production of automated and semi-automated planning systems that are able to utilize the kind of knowledge that human experts bring to bear in planning and design tasks. By this we do not mean the detailed, narrowly-scoped knowledge that every expert possesses about his or her domain of expertise, and which is generally understood to be necessary in planning. Rather, we are concerned with broadly applicable knowledge about planning, scheduling, and design in general. It is our contention that the abilities of both human and automated planners depend critically upon the possession and use of such knowledge (Collins, 1989; Collins and Birnbaum, 1990).

The kind of knowledge with which we are concerned is exemplified by the emerging body of theory and practice relating to the design of manufacturing processes that has come to be known as "lean production" or "just-in-time" inventory management (JIT). JIT theory encompasses a powerful set of general planning and scheduling principles. To take just one such principle, consider the notion of *unit scheduling*. The aim of unit scheduling is to produce a regimen in which each task in a production process takes roughly the same amount of time. The advantage of such a system is that it facilitates a smooth flow of inventory through the production process: No sub-task becomes a bottleneck, and hence no downstream processes are idle waiting for input, nor does inventory back up on the factory floor.[2] Moreover, another principle of JIT management, *flexible manufacturing*, is also facilitated by unit scheduling: Since every sub-task takes roughly the same amount of time, it is relatively easy to customize a schedule to manufacture a limited-production item entailing a particular sequence of sub-tasks. Thus by constraining each sub-task to take the same amount of time, unit scheduling improves coordination to the point where the production process as a whole is both more efficient and more flexible. Factories run on unit scheduling and other JIT principles have proven vastly superior, both in productivity and

[2] In fact, if it does, that indicates a problem in the process causing the bottleneck. Indeed, on a more sophisticated analysis, this is one of the key advantages of the system, in that a key objective of unit scheduling, as of all JIT techniques, is to expose faults in the system as quickly as possible.

quality of output, to factories run using traditional production control techniques.

Interestingly, a principle almost identical to unit scheduling also shows up in computer hardware design. Because the clock speed of a pipelined processor is constrained by the speed of the slowest operation in its instruction set, an optimally efficient instruction set will be one in which the range of times required to execute different instructions varies as little as possible, all other things being equal. Indeed, this is apparently one of the motivations underlying RISC architectures, in which longer instructions that have traditionally been included in instruction sets are instead implemented in microcode. This in turn greatly simplifies scheduling at the instruction level.

General planning knowledge does not arise only from explicit theories of design, scheduling, and planning—it is often implicit in the practices of human planners. Thus, one approach to uncovering such knowledge is to carefully reconstruct the justifications underlying the decisions made by human planners. To take an interesting recent example from the domain of transportation planning, consider the case of "Desert Express," the daily supply flight carried out from Charleston, South Carolina to Saudi Arabia during Operations Desert Shield and Desert Storm.[3] Since flights to Saudi Arabia could be (and generally were) scheduled on an as-needed basis, what was the rationale for instituting a regularly-scheduled flight of this sort? If anything, such rigid scheduling would appear to *reduce* the planners' flexibility without any obvious compensating advantage. Why then would the planners have made this decision? A careful analysis (see section 3.1 below) reveals a number of plausible justifications, all stemming from a single fundamental planning principle, namely, that the fixing of such a constraint permits a number of optimizations in the overall structure of the transportation system. In order to be able to reason intelligently about a plan like Desert Express, a planner must be capable of understanding such a principle and the optimizations it encompasses. No current planning system is capable of performing such reasoning, primarily because no current planning system is capable of representing anything like the set of concepts involved in doing so.

The above examples show that general planning and design knowledge can play a key role in expert-level human planning in areas as diverse as manufacturing management, computer architecture, and transportation planning. Indeed the same principle can often be applied in domains that superficially appear quite different. Moreover, in place of these examples we could easily have substituted a number of others that we have previously uncovered and analyzed (see, e.g., Birnbaum and Collins, 1988; Birnbaum and Collins, 1989; Collins, Birnbaum, Krulwich, and Freed, 1991). All of these examples point to the conclusion that

---

[3] Desert Express flights also originated in Dover, Delaware, and at one point operated twice daily.

expert human planners rely heavily on general planning knowledge of this sort.[4] If computer planning systems are to attain human levels of expertise—or even just to interact effectively with human planning experts—then they must be endowed with the ability to represent and utilize such knowledge.

2 Advantages of general planning and design knowledge

Although existing planning systems generally represent only the "physics" of the environments in which their plans are to be carried out, they are nevertheless able to construct effective plans in some instances. This raises the question of why it is worth constructing systems that incorporate general knowledge about planning, as we propose. There are five general reasons to do so:

(1) **Optimizability.** Human planners have evolved a great body of knowledge concerning ways in which goal-directed systems may be optimized. The just-in-time approach to production management mentioned above provides many good examples of such knowledge. Other important instances include such notions as *contingency planning, parallelization, standardization, economies of scale,* and so on. These sorts of strategies are both extremely powerful and broadly applicable: Plans in almost any domain can potentially be improved, in many cases quite dramatically, by the application of these general optimizing techniques. A system that is capable of utilizing such general strategies must, first, be able to represent the strategies themselves, and second, represent its plans in terms that are general enough to enable it to determine the applicability of a given general strategy to a particular plan.

(2) **Flexibility.** As current work on plan execution makes clear (see, e.g., Agre and Chapman, 1987; Firby, 1989), a plan that appears to be perfectly sound in principle may require numerous modifications, adjustments, and repairs in practice, due to the unpredictable and dynamic nature of the real world. When a plan must be modified, it is critical that the system responsible for the modification understand, in as much depth as possible, how the plan was intended to work. This includes understanding not only the physical causality that the plan exploits, but also the general strategies upon which it is based. For example, a plan based on the strategy of *divide and conquer* may ultimately be expressed as a sequence of steps that do not explicitly refer to either "dividing" or "conquering." However, if the plan must to be modified in the face of unexpected circumstances, any such modification must take its underlying general strategy into account if we are to have any expectation that it will be successful. In effect, abstract planning knowledge is a way of encoding critical facts about the structure of a plan that might otherwise be

[4] A great deal of other work in planning and memory organization leads to the same conclusion as well; see, e.g., Sussman (1975), Sacerdoti (1977), Schank (1982), Chandrasekaran (1987), and Hammond (1989).

unexpressed in the plan's representation, and thus unavailable when the plan must be adapted to its precise circumstances of operation. Moreover, such modification strategies as *substituting resources, "starting over from scratch,"* and so on, themselves provide a good example of general planning knowledge.

(3) **Learnability.** Ideally, a planning system will learn from its mistakes. Such learning will only be useful, however, if the lesson learned can be retrieved and applied when it will be helpful. Learned knowledge must therefore be indexed in a way that facilitates its retrieval when it is relevant (Schank, 1982; Kolodner, 1987; Hammond, 1989). Generality is a critical issue in such indexing, since a lesson learned too narrowly will be of narrow utility. In order to index lessons about planning in such a way that they can be applied generally, the system must possess a vocabulary for describing plans that is itself sufficiently general; this is crucial if the system is to achieve *cross-domain transfer* of learned knowledge (see, e.g., Birnbaum and Collins, 1988), that is, the application of a lesson learned in one planning domain to a problem encountered in a different domain. In effect, this would mean that the system was able to learn and apply its own general principles of plan optimization.

(4) **Comprehensibility.** The three motivations presented above for utilizing general knowledge about planning derive primarily from a concern with improving the performance of automated planning systems. Realistically, however, there is little chance that a computer program will be able to function as a completely autonomous planning agent in a complex, real world domain in the near future. It seems much more likely that advances will come through work on systems that operate in conjunction with human planners. In such systems, *comprehensibility* is a critical issue: The human planner must be able to understand what the automated planner is doing, and why. Existing planners work in much the same way as automated chess-playing programs do, in that they carry out a massive search process, ultimately resulting in a sequence of steps that projection shows is likely to be effective. In chess, a program may discover a sequence of moves that carries out a fork, a trap, a skewer, or any other general chess tactic, without any explicit representation of the fact that the sequence corresponds to such a tactic, or even any intention to carry out such an approach. Similarly, general purpose planners may plan routes or stack blocks without representing anything about the general mechanisms being employed. This makes it difficult for human operators to understand the rationales for the resulting plans.

(5) **Advisability.** This is the converse of comprehensibility: Since human planners often express and justify their plans in terms of general planning knowledge, a computer system that is capable of accepting advice in the form of partially specified plans, general desiderata on such plans, or fruitful

approaches, must be capable of representing and utilizing the terms employed in this way.

3 Case studies in the use of general planning and design knowledge

In this section we will examine in greater depth three case studies in the use of general planning and design knowledge, drawn from three disparate domains: transportation scheduling, engineering design, and counterplanning against threats.

3.1 Transportation scheduling: The case of "Desert Express"

As we mentioned earlier, general planning knowledge is often implicit in the practices of human planners. Consider again the case of "Desert Express." The question of interest here is, why does it make sense to have established such a regularly scheduled service, rather than simply scheduling flights on an as-needed basis? How was overall performance improved?

In fact, a number of system-wide optimizations in transportation planning were made possible by implementing Desert Express, all based on exploiting the fact that, having fixed a constraint, the planner has, by definition, gained information about the system it has so constrained:

- The transportation specialists reduced the demands on their own time, since the planning overhead involved in deciding when a flight should be scheduled and where it should originate could be amortized over multiple shipments. In other words, *regularity amortizes planning overhead* (Wilensky, 1983).

- The knowledge that such a flight would occur regularly permitted planners responsible for the shipment of specific equipment and supplies to Saudi Arabia to plan further ahead with less uncertainty. In other words, *regularity reduces uncertainty about the future.*

- The regularity of the Desert Express flight facilitated the synchronization of other shipping activities with it. For example, shipments feeding into Charleston to be sent aboard Desert Express could themselves be scheduled optimally. In other words, *regularity facilitates synchronization.*

- Both planners and consumers with small but highly time-sensitive shipments might have refrained from requesting the special allocation of transportation assets to ensure timely delivery, since the cost could not be justified in such cases. However, with the institution of Desert Express, such users would no longer hesitate since they knew that the cost of the flight would be incurred in any case: The cost/benefit ratio of express shipment of small items could be more accurately assessed by individual planners in the transportation system

once the system-wide costs and benefits of devoting certain fixed resources to the shipment of such items was made clear. In other words, *regularity makes global decisions about costs and benefits manifest to local decision-making.*

From this example we can draw a number of conclusions about planning in general. At the most abstract level, fixing a constraint of any sort will tend to improve the planner's ability to predict the future course of its own plans, which will in turn enable some optimization. The value of such optimization must be balanced against inefficiencies caused by the loss of flexibility implicit in the adoption of the constraint. More specifically, the transportation planners in this case fixed a constraint on their future plans, in the form of a commitment to carry out a certain kind of action in a certain way on a regular basis. Obviously, it would have made little sense for the planners to consider committing themselves to such an action if it could not have been expected to achieve many of their future goals. For example, the transportation experts could have decided to fly a plane to Mexico City daily, but the extra predictability gained by such a commitment would have been useless for the simple reason that, however predictable, this action would have been useless in furthering their future goals. In order to recognize a useful opportunity to form a commitment to regularly carry out some action, a planner must be able to predict that its future behavior is likely to include the execution of a series of similar plans, all of which might be subsumed by a single, repeated plan. In this case, it was clearly to be expected that a great many supply flights would be operated at various times and from various places in the United States to Saudi Arabia, thus making it at least plausible that a regularly scheduled flight of appropriate frequency could achieve most of the goals of these various flights, while at the same time permitting the optimizations enumerated above.

In order to be able to intelligently consider the construction of a plan like Desert Express, a planner must be capable of carrying out something like the reasoning described above. This in turn means that the planner must be able to reason meaningfully about constraints, flexibility, the value of future knowledge, the predictability of its own future behavior, and the extent to which one plan can be expected to subsume the goals of another. No current planning system is capable of performing such reasoning. This inability, once again, is largely attributable to the fact that no current planning system is capable of representing anything like the set of concepts required in order to do so.

3.2 Design: Coffey stills and turbochargers

Engineering design is another area in which general planning and design knowledge appears to play a key role in expert-level human performance. Engineering design is particularly interesting in that there is an extensive and reasonably well-defined set of general terms that span the various engineering disciplines, including such concepts as *input* and *output, filtering, oscillation* and *damping, mechanical advantage,* and so on. Since human engineers describe

devices in these terms, there is an obvious motivation for making computerized design aids understand them as well. In addition, capturing these concepts in a representation vocabulary will enable a system to formulate and utilize general principles of engineering design, which we have referred to elsewhere as *engineering design themes* (Birnbaum and Collins, 1989). We believe that such general principles of engineering design are coextensive with the general principles of planning: In essence, design is planning with respect to the behavior of a set of artifacts. In fact, many forms of planning that involve the building of large systems, e.g., transportation and manufacturing planning, can equally well be regarded as design problems.

(a) **The Coffey still.** The *Coffey still* (invented about 1830, and named for its inventor) is a device for continuously distilling alcoholic spirits. It consists of two large columns: an *analyzer*, which serves to separate alcohol from substances with a higher boiling point, and a *rectifier*, which serves to separate it from substances with a lower boiling point. In the analyzer, the input liquid—either wine or *wash*, which is basically strong beer—is continuously exposed to steam in order to vaporize its alcohol (and unavoidably, everything else with a lower boiling point as well). The resulting vapor is introduced into the rectifier in order to condense the alcohol and separate it from other substances contained in the vapor. Pipes carrying the cold wine or wash on its way to the analyzer pass through and around the rectifier to cool the vapor. The vapor rises in the rectifier, becoming cooler the longer it is exposed to the cooling pipes. Hence, a temperature gradient exists in the rectifier, and, once equilibrium is reached, various distillates, including alcohol, can be drawn off from the locations in the rectifier corresponding to their condensation temperatures.

(b) **The liquid fuel rocket engine.** In a liquid fuel rocket engine, liquid oxygen and a fuel—typically liquid hydrogen—are pumped into a combustion chamber where they are ignited. On its way to the combustion chamber the liquid oxygen is first passed through a pipe that is flush with, and coiled around, the nozzle of the rocket, which cools the nozzle to prevent it from melting, while at the same time warming the oxygen.

(c) **The turbocharger.** A *turbocharger* is a device to compress the intake air of an internal combustion engine. Since compressed air contains more oxygen in a given volume than uncompressed air, turbocharging permits the combustion of a greater amount of fuel than would otherwise be the case. This results in more powerful explosions upon combustion, which in turn drives the pistons harder. The turbine compressing the intake air is driven by a shaft connected to another turbine that is, in turn, driven by the exhaust gases being forced out of the cylinders by the pistons on the fourth stroke of the cycle.

Figure 1: Device descriptions for the Coffey still, liquid fuel rocket engine, and turbocharger.

Let us consider briefly an example taken from an earlier paper of ours investigating issues in case-based reasoning (Birnbaum and Collins, 1989). In this paper, we pointed out that a number of engineering artifacts, including Coffey stills, rocket engines, turbochargers, and jet engines, share a common design theme, which we labeled *utilizing otherwise wasted output energy to preprocess the input*. In a Coffey still (see figure 1a), hot vapor produced by the analyzer is cooled in the rectifier by transferring heat to the liquid entering the analyzer, with the beneficial side-effect of warming this liquid and reducing the

energy needed to heat it once it enters the analyzer. In a rocket engine (see figure 1b), a similar process occurs, with the nozzle that directs the hot exhaust gases being cooled by a flow of liquid oxygen on its way to the combustion chamber. A very different design illustrating the same general principle is the use of a turbocharger to increase the efficiency of an internal combustion engine (see figure 1c). A turbocharger consists of a turbine that is driven by the exhaust gases exiting the engine, which in turn drives a compressor for the air that is about to enter the engine's combustion chambers. As a result of this harnessing of energy that would otherwise have escaped through the exhaust system, the density of the intake air is increased, thus in turn increasing the amount of oxygen that can be forced into the cylinders, and, ultimately, boosting the output of the engine.

Thus, although these three devices have vastly different functions, and operate on different physical principles, on a deeper analysis they nevertheless exhibit strikingly similar underlying structures, arising from the exploitation of the same general design principle. We believe that such principles form a key element of the knowledge of expert human engineers. Moreover, as we argued above, we expect these principles to overlap significantly with the general principles of planning that form the core of this proposal.

3.3 Counterplanning: The night watchman

As our third case study in general planning and design knowledge, consider the task of *threat detection:* The goal of threat detection is to notice instances of threats against the system's goals early enough to be able to counterplan against them, that is, to do something to prevent them from coming to fruition. Thus the task entails, among other things, *determining "perceptual" features* associated with instances of such threats, *monitoring* the environment for the presence of such features, and *notifying* some decision-making authority when such features are detected. At the next level of detail, determining an appropriate set of features to be monitored entails discovering or generating features that meet certain specifications: The features must *arise early* enough in the course of a threat instance to enable the system to block the threat; they must be reasonably *diagnostic* of threats of a given type, e.g., they must not present the decision-making authority with too many *false alarms;* they must be reasonably *easy to detect;* and so on.

The key thing to notice about the above description of the issues that arise in threat detection is that it is entirely general in nature (Collins, Birnbaum, Krulwich, and Freed, 1991). The concepts involved have nothing to do with detecting threats in any particular domain: Rather they pertain to the task of threat detection in general. This suggests that threat detection, and counterplanning generally, is an aspect of planning behavior that is highly dependent upon general planning and design knowledge. In particular, the ability to modify threat detection behavior in order to successfully meet the

particular circumstances in which the planner finds itself depends crucially upon such knowledge.

Consider the following example: A night watchman completes his rounds in a certain period of time, say one hour. At some point during the night, he discovers that while he was elsewhere on his rounds, one of the areas that he was guarding has been successfully broken into by burglars, and valuable property has been stolen. What should he do to prevent this from happening in the future?

One obvious strategy he might follow is to speed up his rounds. The question of interest here is, what makes this obvious? The answer lies in our general knowledge about the monitoring sub-task in threat detection. Monitoring a feature means focusing the planner's perceptual mechanisms on an area in which a threat might arise and computing whether that feature is in fact present at that time. Since we cannot in general afford to monitor all features continuously, we must instead monitor them intermittently. The rate at which we monitor a given feature must be a function of, among other things, the rapidity with which the threat it predicts can be carried out: A threat that takes a long time to come to fruition need only be checked for occasionally. On the other hand, a threat that can be executed quickly must be monitored at short intervals, in order to ensure that enough time remains after its detection to prevent it from being carried out. Given this understanding of the situation, it can be seen that the speed with which the night watchman makes his rounds is what controls the rate at which threats are being monitored. It is part of our general planning knowledge that if threats are not being detected early enough, one possible repair strategy is to increase the rate at which they are being monitored, which in this case means speeding up the watchman's rounds.

4 Potential applications

In this section we describe three potential applications that critically depend upon the development of representations for, and comprehensive libraries of, general planning and design strategies as described above.

4.1 An intelligent operator editor for planners

The goal of an intelligent operator editor would be to assist human programmers in writing operators for current generation planners. Based on the representation language and library of general planning knowledge, it would be possible to develop dimensions along which operators could usefully be evaluated with respect to optimization (and other) strategies in the library. These would then be used to guide the user/programmer in developing more efficient and effective operator definitions.

We envision the following scenario: First, the user/programmer would enter a preliminary definition of the operator, perhaps along with certain other information about its resource usage, temporal duration, and so forth. Then, based either upon some knowledge of the domain of application, or simply upon information entered interactively by the user/programmer, the system would enter into a dialog with him or her concerning ways in which the operator definition might be improved. For example, the system might make the following sorts of suggestions, where each hypothetical suggestion is preceded by the general planning strategy upon which it is based:

- *Unit scheduling:* "This operator is projected to take approximately 10 minutes to be accomplished. Most of the other operators you have written are estimated to take about 3 minutes each. Could you break this operator up into 3 chunks of roughly equal duration?"

- *Standardization of resources:* "You've defined this operator as requiring a ratchet. No other operator you've defined requires this tool. Could some other tool that you've already specified be used instead?"

- *Resource conflict and parallelization:* "You've defined this operator as requiring a radial-arm saw. Is it likely that it would be used in a plan in conjunction with Operator 17 (...), 19 (...), and/or 35 (...)? They also require the use of the radial arm saw, which would make it impossible to schedule them in parallel with this operator. Could some other tool be used instead?"

- *Interrupt-proof (stable) sub-tasks:* "This operator appears to have a greater number of preconditions than most. Are they stable enough to be achieved serially, and to persist until this operator can be run? Or can the operator be broken up into smaller chunks that require fewer preconditions and result in stable output states?"

By providing guidance in the programming of operator definitions, we believe that an intelligent operator editor of this sort could provide the leverage in adapting planners to new domains that has heretofore been lacking.

4.2 Transportation plan optimizer

As part of an ongoing project, we and our colleagues have conducted a series of interviews with the transportation planning specialists involved in the Desert Shield/Desert Storm operation. The original purpose of these interviews was the production of a library of videotaped "war stories" to be used in an exploratory hypermedia system, Trans-ASK, designed to train novices in transportation planning (Bareiss and Osgood, 1993). However, the data gathered in support of this system could potentially provide the basis for a prototype transportation plan optimization system as well.

A transportation plan optimization system based on our approach would incorporate a set of general design principles for transportation plans, relating to issues such as resource allocation, scheduling, communication, coordination, and contingency planning. The system would take as input a description of a transportation plan, along with a record of the current performance of the plan in operation, when available. These data would then be evaluated with respect to the system's general planning and design principles. Instances in which the plan appeared to diverge from these principles would be flagged for the user, with a suggestion of the type of modification that might be made. For example, a system incorporating the principles discussed in section 3.1 above might suggest situations in which as-needed shipping schedules could be replaced with regular routes. Other potential optimizations include the following:

- *Spotting potential bottlenecks:* During planning, this is generally signaled by a gross mis-match between lift assets and requirements; during execution, it can be spotted downstream by an clear failure to meet requirements, or by underutilized lift assets, and upstream by the piling up of materiel at unexpected locations.

- *Spotting potential synchronization problems:* During planning, this is generally signaled by tight scheduling at a transfer point; during execution, it can be spotted by repeated missed connections.

- *Flagging potential communication problems with transport users:* During planning, this is a potential problem when specifics of time and location cannot be specified until very close to deadlines; during execution, it might arise if changes in time or location occur frequently or close to a deadline.

We do not expect a system of the sort outlined above to be capable of carrying out a detailed analysis of the trade-offs between conflicting planning principles, nor even to incorporate all of the principles that a human expert might be expected to know. Nonetheless, we believe that a system that encompasses even a subset of the principles utilized by human experts in transportation planning could significantly increase the productivity of human planners, especially in a crisis, by quickly highlighting areas of likely improvement.

4.3 Case-based teaching of design principles

Case-based teaching (Schank, 1991) is based on the idea that interesting cases or stories of real examples must be made available to students—or, more proactively, brought to their attention—at just the times at which the contents of those cases are relevant, e.g., when they provide useful advice or warnings concerning an activity the student is currently pursuing. We believe that the representation language and library of general planning and design knowledge envisioned here could play a key role in facilitating the development of case-based teaching systems by providing a rich indexing language, as well as an

important set of teaching points. That is, it is our hypothesis is that much of what needs to be taught in a domain such as engineering design, or for that matter, transportation planning, is how to apply general planning and design knowledge in such a setting.

Our approach to the construction of a case-based teaching system for engineering design would exploit the representation language and library of general planning and design knowledge to facilitate the construction of a relatively course-grained model of the design process. Interesting cases of success or failure would then be indexed at appropriate places in the model, resulting in a well-indexed case base that could be used for browsing in a hypermedia system.[5]

The scenario we envision is as follows: A student would be given a well-documented, finished design, and asked to modify it to meet a set of new requirements. To take a concrete example, the student would be asked to redesign a computer-controlled shock absorber, originally designed for use in a subcompact car, for use in a considerably heavier off-road vehicle.[6] The student's first thought would be, presumably, that the shock absorber does not have sufficient capacity to do the new job.

There are a number of things that might be done about *insufficient capacity* in an artifact. One obvious general strategy is to simply use *multiple instances* of the original design. However, the use of this strategy raises the following question: Should the shock absorbers be connected in *series*—that is, should one shock absorber be attached to the frame of the car, and the second one attached to the *first*, and then to the axle—or should they be connected in *parallel*— i.e., both attached to the frame and to the axle, but not to each other? These notions, and this set of choices, is entirely general in nature: They arise in any situation in which multiple instances of a device are being used in an effort to overcome their insufficient capacity when used singly.

What are the advantages and disadvantages of each approach? Indexed under *series* we might find the following true story: A number of years ago, one of the worst civil engineering disasters in U.S. history occurred in a Kansas City hotel. The hotel had a large, multi-story lobby through which passed two suspended walkways. The walkways, jammed with revelers during a holiday party, collapsed, and over 100 people were killed. The subsequent investigation uncovered what had happened: In the original design for the walkways, a set of supports descended from the roof, passed through the upper walkway, and terminated at the lower walkway. Thus both walkways were suspended *in*

---

[5] A number of systems along this line have already been developed with some success (see, e.g., Ferguson, Bareiss, Birnbaum, and Osgood, 1992).

[6] The redesign scenario, including the particular example we use here, is based on a project being developed by Dr. Catherine Baudin and her colleagues at NASA Ames Research Center and Stanford University's Center for Design Research (Baudin, Gevins, Baya, and Mabogunje, 1992).

parallel from the roof. Unbeknownst to the original designers, however, an on-site modification was made to this design, and as actually built, the supports descending from the roof terminated at the upper walkway; a few inches away, a *separate* support descended from the upper walkway to the lower walkway. In other words, the walkways were now suspended *in series*. The problem with this design, of course, is that the upper walkway now had to support not only its own weight, but the weight of the lower walkway as well. Despite this gross error in design, the skyways managed to stand for a number of years until, overloaded with people, they collapsed.

Confronted with this story, the student's task would be to determine whether the circumstances it describes, or analogous ones, obtain in the current situation. The key point is that by use of the appropriate general vocabulary, it is possible to present interesting, relevant cases at the appropriate time.

5 Conclusions

We believe that the critical issue in planning at this time is to place representation on an equal footing with architecture by articulating and taxonomizing the general planning and design knowledge of human planners. Such an approach offers a number of benefits:

- The incorporation of general planning knowledge will significantly improve system performance on a variety of tasks in a diverse set of problem domains.

- Because human planners clearly think and talk in terms of such general planning knowledge, appropriate representations of this knowledge will, we believe, play a key role in the development of a new generation of automated tools, job aids, and teaching systems that are both more effective and substantially easier to use.

- For similar reasons, we believe that the resulting language for describing planning decisions and their justifications will not only facilitate interactions between human planners and computer systems, but between different computer systems as well—in other words, that it is a necessary precondition for the development of systems for controlling, coordinating, and communicating information between different planning systems utilizing a spectrum of approaches to tackle a wide variety of problems.

In sum, we believe that focusing our efforts on this area, which has clearly not received the attention that it deserves, offers the best chance of advancing the state of the art in computer planning systems.

We conclude with a brief discussion of one particularly interesting application area for general planning knowledge. A key problem in any large decision-

making system is coordinating the efforts of multiple agents, be they human or machine. This is especially true when the different agents are specialized to perform particular tasks employing special-purpose techniques. Coordination and communication then become difficult simply because the agents speak different languages; and they speak different languages because they conceptualize the world in different ways. To effectively combine the efforts of these different agents thus entails developing a common language—or to be more precise, a common ontology—that can be used to structure their interactions.

Eventually, such an ontology would enable individual planning subsystems to explicitly represent and reason about their purposes within the planning system as a whole, the purposes of other subsystems, the strengths and weaknesses of different problem solving methods, and so on. This in turn would enable them to ascertain, for instance, where to turn for advice on a particular problem, or whether some query were within their competence to answer.

Much long-term research remains to be done before such a vision can be realized. However, we believe that a number of tangible medium- and even short-term improvements in our abilities to coordinate the efforts of multiple planners working on the same problem are quite feasible. For example, the categories of planning issues embodied in a representation of general planning knowledge could be used to structure queries and responses in an electronic mail system between human planners in a networked workstation environment. This in turn would permit the development of such services as automatic routing and re-routing of messages to appropriate recipients, prioritization of queries, or even the automatic retrieval of messages relevant to a particular planning task.

Acknowledgments: We thank Ray Bareiss for many useful discussions on these topics. This work was supported in part by by the Air Force Office of Scientific Research under grant no. AFOSR-91-0341-DEF, and by the Advanced Research Projects Agency, monitored by the Office of Naval Research under grant no. N00014-91-J-4092. The Institute for the Learning Sciences was established in 1989 with the support of Andersen Consulting, part of The Arthur Andersen Worldwide Organization. The Institute receives additional support from Ameritech and North West Water Plc, Institute Partners.

References

Agre, P., and Chapman, D. 1987. Pengi: An implementation of a theory of activity. *Proceedings of the 1987 AAAI Conference*, Seattle, WA, pp. 268-272.

Bareiss, R., and Osgood, R. 1993. Applying AI models to the design of exploratory hypermedia systems. *Proceedings of the Fifth Annual ACM Conference on Hypertext*, Seattle, WA.

Baudin, C., Gevins, J., Baya, V., and Mabogunje, A. 1992. Dedal: Using domain concepts to index engineering design information. *Proceedings of the Fourteenth Cognitive Science Conference*, Bloomington, IN, pp. 702-707.

Birnbaum, L. and Collins, G. 1988. The transfer of experience across planning domains through the acquisition of abstract strategies. In J. Kolodner, ed., *Proceedings of the 1988 Workshop on Case-Based Reasoning*, Morgan Kaufmann, San Mateo, CA, pp. 61-79.

Birnbaum, L. and Collins, G. 1989. Reminding and engineering design themes: A case study in indexing vocabulary. In K. Hammond, ed., *Proceedings of the 1989 Workshop on Case-Based Reasoning*, Morgan Kaufmann, San Mateo, CA, pp. 47-51.

Chandrasekaran, B. 1987. Towards a functional architecture for intelligence based on generic information processing tasks. *Proceedings of the Tenth IJCAI*, Milan, Italy, pp. 1183-1192.

Chapman, D. 1987. Planning for Conjunctive Goals. *Artificial Intelligence*, vol. 32, pp. 333-337.

Collins, G. 1989. Plan creation. In C. Riesbeck and R. Schank, eds., *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 249-318.

Collins G. and Birnbaum, L. 1990. Problem-solver state descriptions as abstract indices for case retrieval. *Working notes of the 1990 AAAI Spring Symposium on Case-Based Reasoning*, Stanford CA, pp. 32-35.

Collins, G., Birnbaum, L., Krulwich, B., and Freed, M. 1991. Plan debugging in an intentional system. *Proceedings of the Twelfth IJCAI*, Sydney, Australia, pp. 353-358.

deKleer, J. 1986. An assumption-based TMS. *Artificial Intelligence*, vol. 28, pp. 127-162.

deKleer, J., and Brown, J. 1985. A qualitative physics based on confluences. In J. Hobbs and R. Moore, eds., *Formal Theories of the Common Sense World,* Ablex, Norwood, NJ.

Ferguson, W., Bareiss R., Birnbaum, L. and Osgood, R. 1992. ASK systems: An approach to the realization of story-based teachers. *Journal of the Learning Sciences,* vol. 2, pp. 95-134.

Fikes, R., and Nilsson, N. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence,* vol. 2, pp. 189-208.

Firby, R. 1989. Adaptive execution in complex dynamic worlds. Technical report YALEU/CSD/RR #672, Yale University, Dept. of Computer Science, New Haven, CT.

Forbus, K. 1984. Qualitative process theory. *Artificial Intelligence,* vol. 24, pp. 85-168.

Forbus, K. 1990. The qualitative process engine. In D. Weld and J. deKleer, eds., *Readings in Qualitative Reasoning about Physical Systems,* Morgan Kaufmann, San Mateo, CA, pp. 220-235.

Hammond, K. 1989. *Case-Based Planning: Viewing Planning as a Memory Task.* Academic Press, San Diego, CA.

Kolodner, J. 1987. Capitalizing on failure through case-based inference. *Proceedings of the Ninth Cognitive Science Conference,* Seattle, WA, pp. 715-726.

Kuipers, B. 1986. Qualitative simulation. *Artificial Intelligence,* vol. 29, pp. 289-338.

McAllester, D., and Rosenblitt, D. 1991. Systematic non-linear planning. *Proceedings of the Ninth AAAI Conference,* Anaheim, CA, pp. 634-639.

Sacerdoti, E. 1977. *A Structure for Plans and Behavior.* American Elsevier, New York.

Schank, R. 1982. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People.* Cambridge University Press, Cambridge, England.

Schank, R. 1991. Case-based teaching: Four experiences in educational software design. Technical report #7, Northwestern University, The Institute for the Learning Sciences, Evanston, IL.

Sussman, G. 1975. *A Computer Model of Skill Acquisition*. American Elsevier, New York.

Wilensky, R. 1983. *Planning and Understanding*. Addison-Wesley, Reading, MA.

Wilkins, D. 1988. *Practical Planning: Extending the Classical AI Paradigm*. Morgan Kaufmann, San Mateo, CA.