This paper appeared in the proceedings of AAAI-98.

Toward Design as Collaboration

Susan L. Epstein

Department of Computer Science
Hunter College and The Graduate School of The City University of New York
New York, NY 10021 USA
epstein@roz.hunter.cuny.edu

Abstract

In design, multiple disparate goals must be addressed simultaneously. It is the thesis of this work that problems in two-dimensional layout design can be solved by collaboration among single-goal, intelligent agents, each responsible for a class of objects and responsive to explicit metrics. In this model, each agent produces conflict-free designs for its own class of objects, and then, when objects conflict with each other in the combined design, the agents that own those objects address the conflicts. A limitedly rational implementation demonstrates its efficacy for park layout design in the two-dimensional plane.

Two-dimensional Layout Design

Design problems typically entail large search spaces and multiple, ill-defined goal tests (Goel and Pirolli 1989). As a result, design has been regarded as a domain (CAD/CAM) in which computers assist people rather than work alone. This paper's primary contributions are a model for autonomous two-dimensional layout design as collaboration among a set of agents, and a limitedly rational architecture for this model. Drawing upon research on human experts, the model devotes considerable effort to the selection of an initial high-quality state likely to include multiple constraint violations, and then seeks to remove them.

The problem in two-dimensional layout design is to position a set of two-dimensional objects within a prespecified outline to meet a set of restrictions (*criteria*). In the park design problem of Table 1, for example, 13 objects must be located precisely on a grid. Criteria that must be satisfied are called *constraints*. In Table 1, only the object criteria and C_5 are constraints. Table 1's task is non-trivial; about 1.3×10^{26} placements abide by the object constraints.

A *solution* to a two-dimensional layout design problem positions all the objects and satisfies all the constraints. Typically there are many solutions; it is the multiple, vague goal tests make design problems particularly difficult (Goel and Pirolli 1992).

Those tests are represented here as non-required criteria called *principles*. (There are 7 in Table 1.) Principles are important to a designer, but in some unspecified combination. As a result, design is not an optimization problem, yet designers speak of solutions that are "better" or "worse" (Goel and Pirolli 1992).

Thus in a

Table 1: A park design problem.

Task: Place on a 20 × 40 discrete grid the following objects:

PF_1: a 7 × 7 playing field	P_1: a 5 × 3 pond	B_1: a 5 × 10 building	F_1: a 6 × 7 forest	R_1: a road of width 1
PF_2: a 5 × 2 playing field	P_2: a 3 × 2 pond	B_2: a 2 × 2 building	F_2: a 4 × 2 forest	R_2: a road of width 1
PF_3: a 5 × 2 playing field				R_3: a road of width 1
				R_4: a road of width 1

Goal: Satisfy all constraints and respond to all principles as well as possible.

Object constraints:

- PF_1 boundary is ≥ 0.1 away from all edges.
- PF_3 center is within 0.8 of the park's center.
- B_1 boundary is against some edge.
- B_2 center is within 0.5 of the park's center.
- P_1 boundary is ≥ 0.3 away from all edges.
- P_2 center is within 0.3 of the park's center.
- F_1 center is within 0.4 of the park's center.
- F_2 boundary is within 0.1 of some edge.
- R_1 runs from the eastern grid edge.
- R_2 runs from the western grid edge.
- R_3 runs from the northern grid edge.
- R_4 runs from the southern grid edge.
- R_1, R_2 , R_3 , and R_4 are connected.

Intra-class principles:

C_1: Total road length should be small.

C_2: Fields should not adjoin each other.

C_3: Ponds' centers should be ≤ .2 of the grid apart.

C_4: Buildings' centers should be ≤ .6 of the grid apart.

Inter-class constraint:

C_5: Objects may not overlap.

Inter-class principles:

C_6: Buildings should be adjacent to roads.

C_7: Every building should have a view, i.e., be within 2 units of a pond or forest.

Secondary inter-class principle:

C_8: Minimize the total Manhattan distance of all empty grid positions to their nearest road.

two-dimensional layout design problem, how well a solution meets all the principles becomes a metric on its quality. The goal is to find a high-quality solution.

Object Categories and a Collaborative Model

Two-dimensional layout design problems usually categorize the objects to be instantiated (Goel and Pirolli 1992). Table 1, for example, partitions its objects into five *classes*, sets of functionally and/or structurally similar objects: ponds, playing fields, forests, buildings, and roads. (The remainder of the park is intended to be open grassy space.) Design criteria can be categorized with respect to these classes. An *object criterion* describes how a single object relates to the grid, such as "the boundary of B_1 is against some edge." An *intra-class criterion* restricts two or more objects of the same class. For example, in Table 1 the intra-class criterion "Total road length should be small" describes a desirable property for the set of roads. An *inter-class criterion* restricts two or more objects of different classes. For example, in Table 1 the inter-class criterion "Buildings should be adjacent to roads" describes how each building should relate to some road.

To cast two-dimensional layout design as a state space search, let a description that positions every objects be a *complete state*; otherwise it is a *partial state*. In addition, let a state be *legal* if it violates no constraints; otherwise it is *illegal*. A two-dimensional layout design problem can then be addressed as search for a path from a partial, legal state through a space of legal states to a complete legal state. Using this single-goal approach, a designer would locate one object at a time on the grid until all objects were present, and backtrack as necessary. (Constraint satisfaction programming, *CSP*, is addressed in the final section.)

Human design experts, however, do not search this way. They consider a set of objects or parts of objects and how they relate to each other, typically in an illegal state (Schraagen 1993). In addition, a human expert's first state is likely to be a compendium of a number of legal partial states, each for an entire class of objects

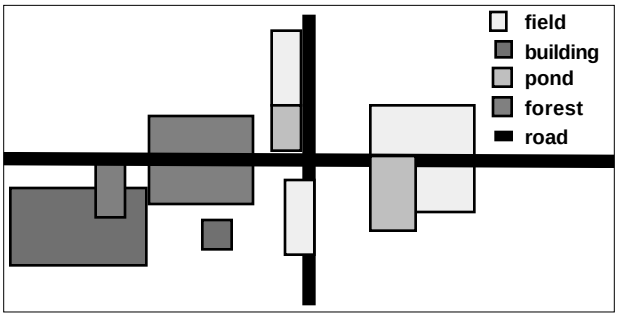


Figure 1: An initial state with 6 conflicts for Table 1.

Since there are usually many different possible states, the human designer selects, both initially and during search, states that meet metrics for good design. A metric applicable to legal partial states and limited to a single agent's objects is an intra-class principle. A metric applicable to complete states and pertaining to objects of more than one agent is an inter-class principle.

In light of the human approach, this paper proposes a collaborative model that addresses two-dimensional layout design as a task for a set of agents, each of which advocates on behalf of a single object class. For example, five separate agents would collaborate on the problem in Table 1. Problem solving in this model is search for a path through a space that also includes both illegal and partial states. This collaborative model for two-dimensional layout design begins search from an initial state that respects the principles of each agent. Each agent produces an *ideal framework*, a legal partial state for all its objects that is highly-rated by its principles. For Table 1, where C_1 is the only all-road principle, one ideal road framework is four orthogonal roads from the center of the grid to the edges. The union of the agents' ideal frameworks is a complete, initial state, such as Figure 1, and likely to be illegal.

Since agents produce legal partial states, a *conflict* arises in this model only when two or more agents violate an inter-class constraint, as when objects overlap in Figure 1. A conflict is uniquely identified by its set of objects. There are 6 distinct conflicts in Figure 1: between the large field and the large pond, the large field and the eastern road, the small forest and the large building, the large forest and the western road, a small field and the southern road, and the strip contested by the large pond, the large field, and the eastern road.

Collaboration among agents in this model can be either explicit or implicit. Explicit collaboration would be negotiation over a conflict. Implicit collaboration is achieved by the influence of shared inter-class principles on decision making. For now, the latter is the primary focus. Consider, for example, the conflict between the large building and

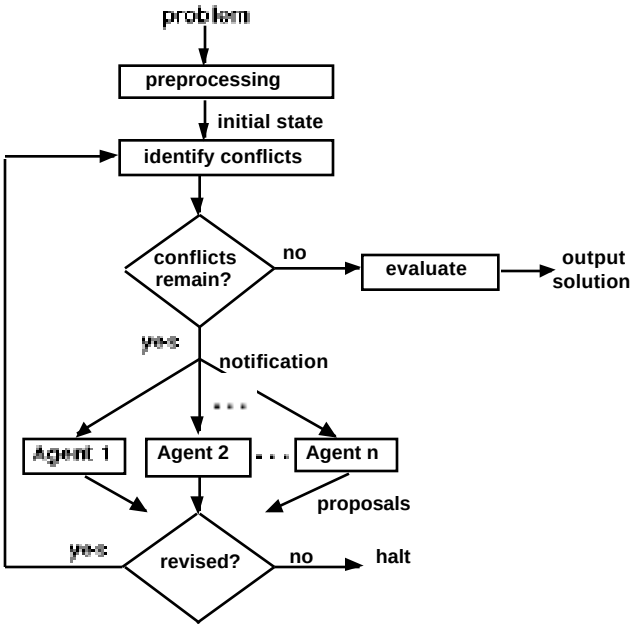


Figure 2: An overview of the CD solution process.

the small forest in Figure 1. One way to resolve it is to relocate either or both objects. When the forest agent considers a forest relocation, it will prefer those that abide by C_7. The building agent, meanwhile, will prefer building relocations that abide by C_4, C_6, and C_7. (Consideration of the secondary principle C_8 is postponed, as discussed later.) Because the strength with which each agent suggests a relocation reflects how well that proposal meets the agent's principles, the strongest suggestion represents a decision that best addresses both mutual and individual goals.

The CD Architecture and FLO

CD (Collaborative Design) is a limitedly rational architecture which applies the collaborative model to two-dimensional layout design problems. The CD program for park design is called *FLO*, after park designer Frederick Law Olmstead. Although CD is not restricted to parks, for clarity the examples here are all drawn from runs with FLO. A CD-based system begins with partially specified objects already categorized into classes. CD's task is to specify them fully. For example, Table 1 restricts each object's location in the grid; a solution locates all of them precisely.

CD assigns the full specification of the objects in any one class to a limitedly rational agent. The current problem state describes the objects' specifications, and represents the agent's environment. Each agent computes and executes responses to that environment in the form of proposals that specify its own objects further or differently and abide by constraints.

Table 2: Procedures available to a CD agent

Type	Role
Originator	Propose conflict resolution
Commentator	Evaluation metric
Secondary commentator	High-cost evaluation metric

Figure 2 sketches the CD solution process (*trial*). When a problem is first presented, CD constructs an initial state. Objects that conflict there notify their respective agents. An agent whose object is in conflict with another's tries to construct *proposals*, recommendations that address the conflict. A proposal includes the agent's name, the conflict it addresses, the action it proposes, and comments on how well the proposal abides by the agent's principles.

Until some state is conflict-free or there are no proposals, CD selects the proposal that has the best comments and addresses the most *severe* conflict. In FLO, that is the conflict that involves the most objects and the most grid locations, and engenders the fewest proposals. Implementation of the selected proposal creates a new state, and the cycle in Figure 2 repeats. CD is non-deterministic because ties among equally good choices are made at random. Typically, therefore, CD runs multiple trials, and all the agents evaluate each output solution.

Throughout a trial, each CD agent has access to three kinds of procedures, listed in Table 2. An *originator* is restricted to a single agent and formulates proposals to address a conflict in which its agent's object is involved. A *commentator* evaluates how well the state resulting from a proposal will satisfy a particular metric. A commentator restricted to a single agent represents an intra-class principle; a commentator shared among a set of agents represents an inter-class principle. FLO's road agent, for example, has a C_1 commentator to support short roads, and a C_6 commentator (shared with the building agent) to encourage roads near buildings. A *secondary commentator*, such as C_8, also applies a metric to a state, but one so costly that its application is restricted to preprocessing and solution evaluation only. The remainder of this section details the construction of the initial state and proposal generation.

The initial state

CD devotes substantial effort to construction of an initial state, as diagrammed in Figure 3. To begin, each agent produces several ideal frameworks, descriptions including only that agent's objects and their locations. In this phase, an agent considers only its own objects and abides by all relevant constraints. The agent rates each of its ideal frameworks with its commentators of both kinds, and forwards the top-rated ones.

CD uses the best ideal frameworks from each agent to produce several complete states. Each is a *combined ideal*, the union of one ideal framework from each agent. Figure 1 was a combined ideal, the union of an ideal framework for Table 1 from each of FLO's five agents.

The combined ideal with the fewest conflicts is called the *candidate.* In a candidate, any agent that has more than *x*% of its resources in conflict is labeled an *offender*. For example, all but the building agent are 25% offenders in Figure 1. From its store of ideal frameworks, each offender offers several highly-rated alternatives. CD replaces the offender's ideal framework in the candidate with whichever alternative produces the fewest conflicts. For example, FLO substituted another forest ideal framework in Figure 1 to produce the new combined ideal in Figure 4, with 5 conflicts instead of 6. Every offender is given only one opportunity to resubmit ideal frameworks. CD uses the initial state only as a good starting point, and then goes on to address the remaining conflicts differently.

Proposal generation and limited rationality

As in Figure 2, until the current state is a solution, objects notify their agents when they are involved in one or more conflicts. Each notified agent formulates a set of proposals to address those conflicts. An agent may propose either to *resolve* (eliminate) a conflict between two objects or to *ameliorate* (reduce the number of objects in) a conflict among three or more objects. In the state of Figure 4, for example, two agents reacted to the conflict involving a small field and the southern road. The field agent proposed relocating the field so that it did not conflict with any other

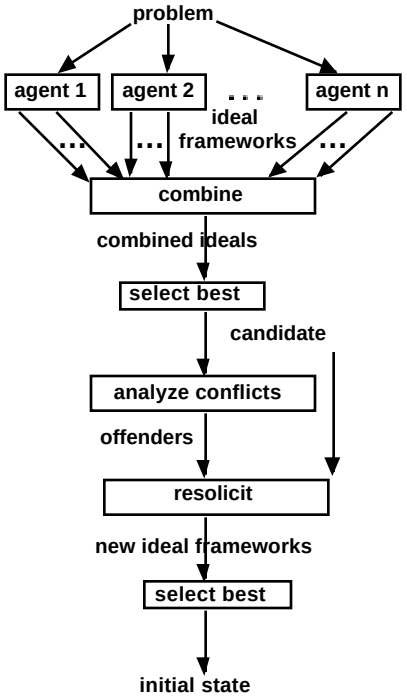


Figure 3: Initial state construction in CD.

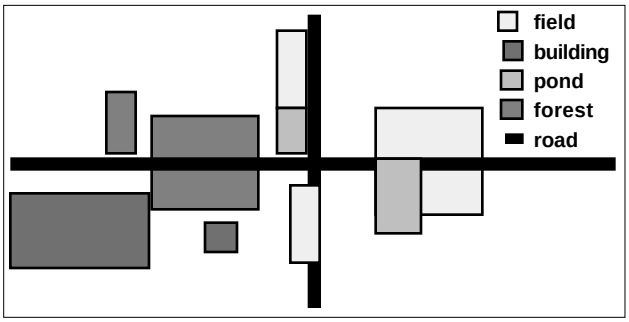


Figure 4: A revision of Figure 1 with 5 conflicts.

object, and the road agent proposed either relocating the southern road or diverting it around the field. Conflicts among more than two agents may be ameliorated first, and then resolved. For example, the pond agent might propose to ameliorate the three-agent conflict over the strip in Figure 4 by relocating the pond. If that proposal were implemented, the large field would then conflict only with the eastern road, and, in the next cycle, the road agent and the field agent could address that reduced conflict. CD always prefers resolution, however, to amelioration.

There are usually more ways to resolve or ameliorate a conflict than a design system should explore. To control the number of proposals, a CD originator suggests only constraint-abiding actions that involve its own objects. For each agent, FLO has one originator that relocates one of that agent's objects. For example, in Figure 4 the western road could shift to any of the other 19 rows. All but 5 of those, however, would still conflict with some other object. Rather than propose many substandard relocations, the road agent's originators enforce relevant constraints. Thus in Figure 4 any road relocation proposal would shift the western road to a clear row and connect it to at least one other road. An originator is forbidden to propose an action that would return to an equivalent prior state. Each agent may have any number of originators. In addition to its five relocators, FLO has a road originator that reroutes a specific road around a specific object.

On two-dimensional layout design problems, even constraint-abiding originators are likely to generate far too many proposals. Therefore CD makes each agent *limitedly rational* by imposing two prespecified limits: one on its total computation time per cycle, and another on the number of proposals it may produce about any one state.

After an agent's originators produce their proposals, its commentators rate how well each prospective new state would meet the agent's principles. Unlike ideal framework construction, comment construction affords the agent a view of the entire state, not just its own objects. Although commentators are not subject to resource constraints the way originators are, care must be taken to guard against system slowdown from those that are resource-hungry. For example, although an efficient implementation discourages roads along the edges of the grid, the commentator for Table 1's C_8 is still costly enough to substantially degrade system performance. C_8 would slow down any system

whenever it was evaluated. It is more economical, therefore, to exclude it during search. Such a secondary commentator is applied with the ordinary commentators only to rate combined ideals during generation ("select best" in Figure 3) and to compare solutions ("evaluate" in Figure 2). Empirically, however, high-quality solutions are less likely as more commentators are made secondary; only the most costly should be relegated to this category.

Empirical Results and Discussion

Several performance measures apply in two-dimensional layout design. The system should be able to address multiple problems. Solutions should be achieved quickly and fit criteria well. Easier problems should be easier to solve. Speed can be measured as elapsed problem-solving (clock) time, or as number of decision cycles.

Table 3: Partial specification of a small park design problem. See Table 1 for intra-class and inter-class principles.

Task: Place on a 10 × 20 discrete grid the following objects:

PF: a 4 × 2 playing field	B: a 3 × 3 building	R_1: a road of width 1	R_3: a road of width 1
P: a 4 × 3 pond	F: a 2 × 1 forest	R_2: a road of width 1	R_4: a road of width 1

Goal: Satisfy all constraints and respond to all principles as well as possible.

Object constraints:

PF center is within 0.8 of the park's center.	F center is within 0.1 of some edge.	R_3 runs from the northern grid edge.
B center is within 0.5 of the park's center.	R_1 runs from the eastern grid edge.	R_4 runs from the southern grid edge.
P center is within 0.2 of the park's center.	R_2 runs from the western grid edge.	R_1, R_2 , R_3 , and R_4 are connected.

CD and FLO are both implemented in Common Lisp. All combined ideals, initial and intermediate states, plus proposals and ratings, can be made transparent to the user. FLO was tested on both the larger problem of Table 1 and on the small problem partially specified in Table 3. Both problems use the intra-class and inter-class principles in Table 1. Each run consisted of 10 trials on the same problem. The results shown in Table 4 are averaged over 10 runs, where "size" counts complete states conforming to all object constraints. Figure 6 shows the top-rated solution to the small problem; it satisfies all the criteria of Table 1.

Table 4: Results on 10 trials for two problems with FLO.

	Small problem	**Larger problem**
Size	3.9×10^{11}	1.3×10^{26}
Offenders	2.20	1.36
Solutions	9.3	7.3
Cycles to solution	2.1	10.0
Time per trial	7.33 seconds	136.9 seconds

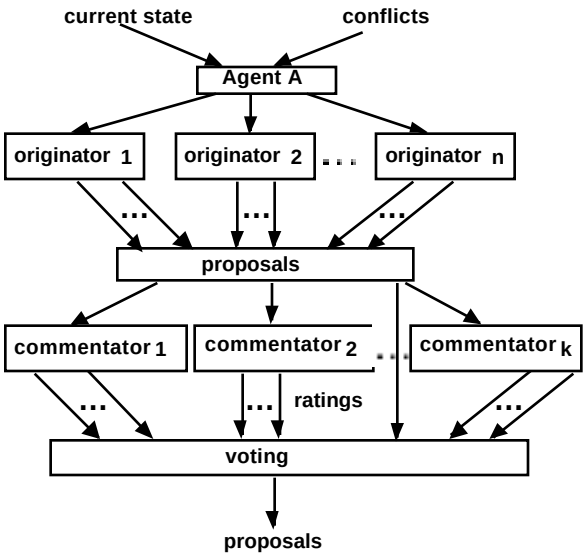


Figure 5: Proposal solicitation from a CD agent.

In a typical trial on the larger problem, the combined ideals ranged from 19 to 6 conflicts; the best was Figure 1, with 4 offenders. Only the forest agent's alternative ideal framework improved that candidate, making the initial state Figure 4 with 5 conflicts. After 4 revisions (the last solved 2 conflicts), FLO arrived at the solution in Figure 7(a), with a road bent around the southernmost field. Another solution from the same run, Figure 7(b), rates lower because the large building has a poor view. The remainder of this section addresses the primary issues in two-dimensional layout design within the collaborative model envisioned here: search, optimality, speed, and generality.

Search

CD executes a satisficing search guided by explicit principles through the space of complete states. CD finds highly-rated solutions so quickly in so large a space because it prunes search several ways. Search begins with a state a highly-rated by the commentators and with the prospect of a short solution path (number of conflicts). The originators prevent searching some predictably illegal states, while limited rationality controls the branch factor. The commen-

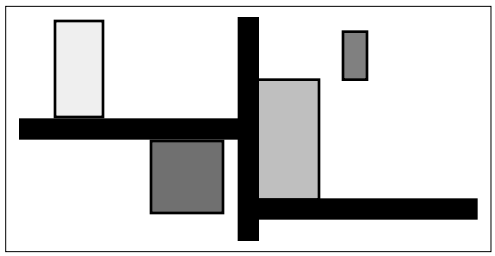


Figure 6: FLO's best solution to the small problem.

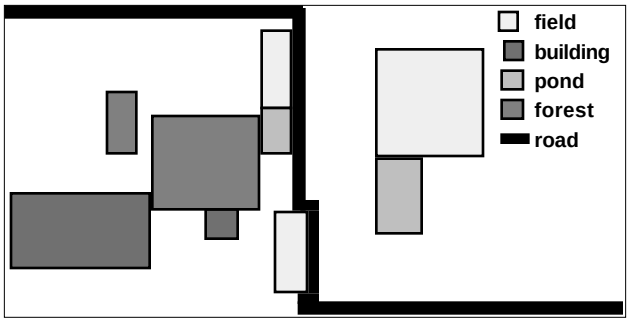
tators bias search in the direction of highly-rated states. Agents proffer only high-rated proposals, and one that promises to resolve some most severe conflict is chosen.

On a finite number O of objects in a finite grid, CD is guaranteed to halt. The system does not cycle through the same set of states (shift one or more objects back and forth repeatedly) because the originators do not make such proposals. In addition, each object is treated as if it has finitely many possible locations on the grid, so that there are finitely many proposals to consider. Let C be the number of conflicts in the initial state, and $obj(c_i)$ be the number of objects involved in conflict c_i. Clearly $obj(c_i) \leq O$ for $i = 1, 2, \ldots, C$. Since CD never introduces new conflicts, and since it either halts or reduces or ameliorates at least one conflict on each iteration, the maximum number of iterations before it halts is:

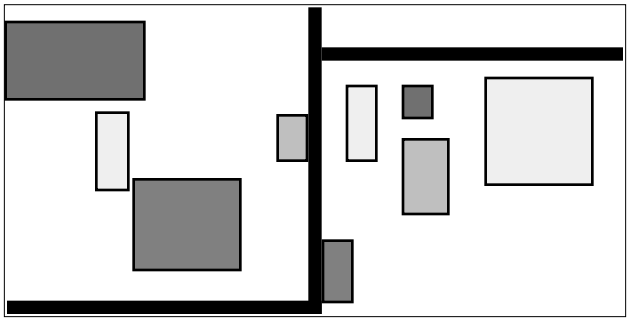
$$\sum_{i=1}^{C} \left[obj(c_i) - 1 \right] < O \quad C$$

OC is an overestimate of the distance to solution, since a single proposal can resolve more than one conflict.

Like many satisficing systems, CD is not guaranteed to find a solution. This is why a run is a set of trials, as it often is in hill climbing. During some trials, an illegal state engenders no proposals, so the program produces no solution. Typically these are "nearly good enough" states with only one or two conflicts remaining. Four approaches to these states are under investigation. First, under limited rationality, a constructive proposal may be overlooked. CD could provide originators with increasing resource limits if a state has only a few conflicts remaining, in a kind of iterative broadening. Second, after the initial state, CD never proposes to introduce additional conflicts. It could allow an originator to introduce a proposal that addresses one conflict at the price of introducing another that is expected to be easier to resolve. Care would be required to preserve halting. Third, CD never violates object criteria, but human designers frequently do (e.g., cost overruns). CD could modify an object's description (e.g., size or shape) within some prespecified limits. Finally, CD's proposal system supports collaboration based on shared principles, but does not provide for direct communication between agents. It could also have agents negotiate a mutual conflict together.



(a)



(b)

Figure 7: Two solutions to the problem in Table 1.

Optimality and high-quality solutions

As discussed in the introduction, the two-dimensional layout designer must satisfy only constraints and may merely "do well" on principles. Since CD does hill-climbing only on constraints, a solution is guaranteed to be optimal only with respect to them. FLO, for example, only guarantees no overlap in a solution to a park problem.

The use of a single non-object constraint in FLO was deliberate. All the principles are goals in design, but making more of them required is not necessarily constructive. That further constrains the generators, and runs the risk of finding too few proposals within the resource limits. Thus there is no guarantee that any commentator will rate an eventual solution highly, only that the solution process will prefer to pass through states the commentator rates more highly than some alternatives, all other decision factors being equal. Enough trials (say 10) seem to guarantee a variety of high-quality solutions from which to select.

Uncertainty about an optimal value for a criterion is characteristic of design problems, but it prevents identification of an optimal design. C_8, for example, is normed on an overestimate; its maximum value is unknown. As a result, solutions can only be better than others, as Figure 7(a) is on C_8, never "best."

Human designers typically tinker with their solutions. In Figure 7(a), for example, lowering the western road one unit is likely to improve the C_8 rating without impacting the other ratings. A tweaking phase is therefore planned for CD, where a solution will be repeatedly improved by hill-climbing through legal states. Additional or refined principles could also further improve the designs of Figure 7.

Speed

Including illegal states substantially enlarges the search space. CD's ideal framework is a highly-rated, albeit illegal, initial state. To test the impact of a refined initial state, FLO was run on the small problem using a single combined ideal without testing for offenders. Compared to the data reported above, there was a 51% reduction in the number of solutions, running time approximately tripled, and the number of iterations to a solution increased by 81%. Clearly, some computational effort on the initial state is worthwhile, but how much does it merit?

In extensive empirical testing, a variety of values for pa-

rameter settings were tried on the small problem, measuring both total computation time and the fraction devoted to construction of the initial state. Number of ideal frameworks generated (1–10), number of combined ideals (1–8), percentage of conflicted resources defining an offender (25-75%), and number of alternatives submitted by an offender (1-5) were tested in many combinations, varying one at a time. In these experiments the initial state almost always needed revision. Up to 6 or 7, more combined ideal frameworks made it more likely that the problem would be solved and that the overall solution time would be low, but it also increased the percentage of total computation time devoted to the initial state. Lower percentages for the offender definition also improved the number of solutions, without any impact on either time. Finally, the number of alternatives submitted by an offender appeared to have little impact as long as it was at least 3. Under the settings used here (10 ideal frameworks, 7 combined ideals, 3 alternatives from 25% offenders), FLO devoted 36% of its computation time to the initial state on the small problem, and 21.3% on the larger problem.

Could FLO have done as well with smaller resource limits on the originators? Table 4's data was generated with limits of 10 seconds and 10 proposals, but some originators (notably the road agent's) used far less. In a run where the originators were held to 5 seconds and 5 proposals, FLO solved the larger problem about as often and the solutions were about the same quality. At the 95% confidence level for statistical significance, however, problem solving required more time (166.0 seconds) because the commentators spent considerably more time evaluating the proposals.

AI research has long recognized that a generate-and-test approach like CD's derives power when intelligence is moved from the tester into the generator. In an earlier version, originators proposed relocations whether or not they produced a conflict, and thereby substantially slowed the program. Forcing the roads to interconnect is a kind of embedded intelligence, too. Additional domain-specific routines (e.g., one that would perceive large chunks of "clear" space quickly and investigate relocation alternatives within them) could further speed performance.

In a parallel implementation, CD agents and shared commentators would be relegated to separate processors, and proposals computed quickly would be considered first. Caution would be advised, however, since a quickly developed proposal is often to a conflict that is easier to resolve, one less worthy of immediate attention.

If the grid is made larger, the number of alternative locations for objects of the same size increases, so that an intuitively easier problem (the same size objects in more space) could require more computation (more proposals to generate and comment upon). CD's limited rationality prevents this. Although such limited rationality could make it more difficult to discover a solution, the quality of FLO's solutions as rated by the commentators does not deteriorate, and it solves problems much faster with these limits in place. Preliminary testing on a still larger problem in a 30×90 grid indicates that FLO should scale nicely.

Two speedup techniques applied here are domain-dependent. First, certain symmetries of the two-dimensional plane produce equally valid object placement for ideal frameworks with relatively little additional computation time. Therefore, FLO symmetrically transforms an offending ideal framework and any proffered alternatives to produce more alternatives quickly. Second, relocators site an object according to its alternatives. Because its object criteria are static, FLO can compute all possible locations for an object as constrained by its object criteria only once, when the problem is defined.

Generality

CD is not limited to parks; it is intended for many two-dimensional layout domains. The design criteria for buildings, for example, also includes object criteria (e.g., room dimensions), intra-class criteria (e.g., "keep the plumbing along a central core" or "provide electrical power to every room,") and inter-class criteria (e.g., "do not place an outlet too near running water"). A variety of other domains is under active consideration.

Within any domain, CD is highly modular. It accommodates any number of object classes and their agents, any grid size, and any number of objects. Each agent may have any number of originators and commentators, each with its own resource limits. Thus, if the solutions are unsatisfactory with respect to some unelucidated principle, it is easy enough to add it.

If, on the other hand, solutions are unsatisfactory with respect to some principles *P* already present, there are several ways to give them more influence. CD normalizes each commentator's ratings in a range before they are submitted. Thus one could bias the system with a higher range for *P*. If, for example, road costs are a particular concern, increase C_1's range. Similarly, CD normalizes the proposal strengths of each agent in a range, so one could assign a higher range to an agent that included *P*. If, for example, playing fields are the park's primary function, increase the field agent's range. Alternatively, CD runs multiple trials on a problem. One could rate those solutions on *P* and identify the best, or further perturb the solution top-rated by *P* in ways that *P* preferred.

This proposed model and the CD architecture are likely to succeed only if the design space meets the following conditions:

- Objects are readily categorizable.
- Each principle quantifies a single aspect of good design.
- Non-secondary commentators are relatively efficient.
- There are easy to find ideal frameworks for each class.
- There are many solutions of varying quality.

These are all typical of two-dimensional layout design space, although other design spaces may also fit this description. If ideal frameworks score equally under some perturbation (e.g., the symmetries applied in FLO), or static object criteria conserve originator time, so much the better.

Related and Future Work

An underlying assumption in CD is that explicit knowledge can control the combinatorics of search in a very large space, an assumption shared by human designers. Although CD is not intended as a full cognitive model of the human design process, it does simulate many features identified by psychological studies of experts in architecture, mechanical engineering, and instructional design (Goel and Pirolli 1992). Like these human experts, CD has distinct problem solving phases (ideal frameworks, initial state, iteration), decomposition into modules (object classes and their agents), incremental development with limited commitment, little deductive inference, and a variety of evaluation methods (CD's commentators and secondary commentators) not traditional in many other AI domains.

CD is a satisficing architecture; like people, it solves problems "well enough" rather than optimally (Simon 1981). There is ample support in the psychology literature for satisficing behavior in a variety of domains and for multiple rationales like CD's commentators (Biswas, Goldman, Fisher, Bhuva, and Glewwe 1995; Crowley and Siegler 1993). Failure-driven conflict resolution, such as the repeated refinements described for CD, has proved successful in a variety of other domains (Collins, Birnbaum, and Krulwich 1989).

Because it requires a group of logically distinct processing agents to address the same problem, CD can be classified as distributed AI, in particular as a multi-agent system (Bond and Gasser 1988). The architecture seeks to realize collective system properties (good design) within a fixed environment (Durfee and Rosenschein 1994). CD is a step toward organizational cognition, where individual agents are responsive both to their own objects and those of other agents (Gasser 1993). CD is also an example of distributed constraint satisfaction, a newly-emerging research area (Armstrong and Durfee 1997). AI in design has received substantial attention (e.g., (Gero and Sudweeks 1996)), but the collaborative model for two-dimensional layouts presented here is novel to the best of the author's knowledge.

Although preliminary indications are that CD and FLO are efficient, a comparison of these results with those of other methods, particularly CSP, is in order. Each FLO object would be represented as a variable whose possible values represented its legal alternative locations. The overlap constraint renders the graph complete. CSP could also play an important role in the formulation of ideal frameworks. Economics-based negotiation is another likely method to consider (Pennock and Wellman 1996). In FLO, the resources are the distinct grid squares for which agents would compete. The author hopes to attract specialists in both areas to these problems. Although other techniques exist, most notably in operations research and numerical methods, their cost for these problems appears prohibitive.

Two of Olmstead's New York City parks (Central Park and Prospect Park, often cited as examples of outstanding design) are intended as further test cases. FLO soon will have originators to rotate objects and to interchange the locations of same-class objects. Several commentators that encourage additional aesthetic properties, such as even distribution of vacant space, are also under development.

CD agents are reactive; they sense conflicts in the grid and respond to them with proposals. An intention shared among agents is thus far represented only as a common decision making principle. An important next step is to enable CD agents to communicate directly with one another. This should be particularly valuable in situations where only joint action (for example, shifting objects of different classes simultaneously) will resolve a conflict.

CD is a direct descendant of FORR, a general architecture for learning and problem solving (Epstein 1994). CD

can be thought of as a set of FORR-based agents, each of which has a set of FORR-like Advisors (the procedures of Table 2). A CD agent, however, has originators so that it is not compelled to comment on all possible actions as a FORR-based agent is. FORR's more sophisticated learning and search approaches should eventually migrate to CD.

Meanwhile, this paper has shown how a collaborative model for two-dimensional layout design can both constrain search and impose aesthetic design criteria. The model capitalizes on collaboration among agents to clarify the process description, to organize search, and to find a good initial state. The CD architecture provides a limitedly rational version of this model which distinguishes between a resource-limited search for good ideas (originators) and good reasons to support them (commentators). CD makes conflict and collaboration transparent to the user, solves easier problems more quickly, and produces a variety of high-quality solutions even though it tolerates constraint violations during search. FLO, the implementation for park design, is fast, modular, and offers a broad set of high-quality solutions.

Acknowledgments

This work was supported in part by NSF grant #9423085, PSC-CUNY #666318. Thanks to Jack Gelfand and the anonymous reviewers for their constructive suggestions.

References

Arm strong, A. and Durfee, E. H. 1997. Dynamic Prioritization of Complex Agents in Distributed Constraint Satisfaction Problems. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence,* 620-625. Morgan Kaufmann.

Biswas, G., Goldman, S., Fisher, D., Bhuva, B. and Glewwe, G. 1995. Assessing Design Activity in Complex CMOS Circuit Design. In P. Nichols, S. Chipman, & R. Brennan (Ed.), *Cognitively Diagnostic Assessment,* Hillsdale, NJ: Lawrence Erlbaum.

Bond, A. H. and Gasser, L. 1988. An Analysis of Problems and Research in DAI. In A. H. Bond, & L. Gasser (Ed.), *Readings in Distributed AI,* 3-35. CA: Morgan Kaufmann.

Collins, G., Birnbaum, L. and Krulwich, B. 1989. An Adaptive Model of Decision-Making in Planning. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 511-516. Morgan Kaufmann.

Crowley, K. and Siegler, R. S. 1993. Flexible Strategy Use in Young Children's Tic-Tac-Toe. *Cognitive Science*, 17 (4): 531-561.

Durfee, E. H. and Rosenschein, J. S. 1994. Distributed Problem Solving and Multi-Agent Systems: Comparisons and Examples. In *Proceedings of the Thirteenth International Distributed AI Workshop*, 94-104.

Epstein, S. L. 1994. For the Right Reasons: The FORR Architecture for Learning in a Skill Domain. *Cognitive Science*, 18 (3): 479-511.

Gasser, L. 1993. Social Knowledge and Social Action: Heterogeneity in Practice. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, 751-757. Chambéry, France: Morgan Kaufmann.

Gero, J. S. and Sudweeks, F. (1996). *Artificial Intelligence in Design '96*. Kluwer Academic Publishers,

Goel, V. and Pirolli, P. 1989. Motivating the Notion of Generic Design within Information Processing Theory: The Design Problem Space. *AI Magazine*, 10 : 19-36.

Goel, V. and Pirolli, P. 1992. The Structure of Design Problem Spaces. *Cognitive Science*, 16 : 395-429.

Pennock, D. M. and Wellman, M. 1996. Toward a Market Model for Bayesian Inference. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, 405-413.

Schraagen, J. M. 1993. How Experts Solve a Novel Problem in Experimental Design. *Cognitive Science*, 17 (2): 285-309.

Simon, H. A. 1981. *The Sciences of the Artificial* (second ed.). Cambridge, MA: MIT Press.