Analyzing a Process of Collaborative Game Design Involving Online Tools

Sandra B. Fan¹, Brian R. Johnson², Yun-En Liu¹, Tyler S. Robison¹, Rolfe R. Schmidt¹, Steven L. Tanimoto¹

1. Department of Computer Science and Engineering
2. Department of Architecture
University of Washington, Seattle, WA 98195, USA
1: {sbfan, yunliu, trobison, rolfe, tanimoto}@cs.uw.edu; 2: brj@uw.edu

Abstract

In this study, we explore how problem solving and design can be modeled using state-space-search methodology, by engaging in the design of two educational games. Additionally, we wanted to discover how online communication tools could be used to support collaborative design. We used three online tools: 1) CoSolve, a collaborative problem-solving environment that we developed, 2) Google Wave, a wiki/chat hybrid communication tool, and 3) INFACT, a discussion forum that we built for use in education. We used these tools to design Go Atom, a chemistry game, and Eco-avelli, a game that intended to demonstrate the politics of climate change. Using our game design experience as an example, we present a method for modeling design processes using state-space-search, describe how the communication tools were used in our process, and present suggestions for ways to use these tools to tackle design problems in the future.

1. Introduction

As web and communication technologies mature, there is an increasing need to discover more effective uses of these technologies. In particular, we are interested in whether the web can be used to support collaborative problem solving and design, and how best to build web tools to do so. Online communication tools have played important roles in collaborative problem solving, and we can expect their roles to grow in importance as new affordances make them more effective, and as more people participate.

We explore this issue by engaging in our own design process, to identify what the advantages and disadvantages of current tools are. We designed two simple, face-to-face, educational games for children, and used online collaboration tools to support our collaboration, with the intention of using our experience to inform our efforts to develop online collaboration tools for problem solving across a range of disciplines. In particular, we take a state-space-search approach to modeling the design process, based on the classical AI theory of problem solving.

During each design exercise, we utilized different online collaboration tools: our own INFACT system, Google Wave beta, as well as our CoSolve environment for modeling state-space problem solving, and we developed two games: Go Atom, a chemistry game, and Eco-avelli, a climate change game.

In Section 2, we will describe the existing technologies and the state-space-search design methodology in more detail. In Sections 3, 4 & 5, we will describe our specific design process of creating these two games. Section 6 describes our state-space model of design as applied to our creation of these games, and presents results of our analysis. Section 7 reports on our usage of the tools and suggestions for how the design process we engaged in could have been improved, and we conclude with general discussion in Section 8.

2. Background

2.1. Communication Tools that Support Online Collaboration

Before describing the particular systems we used in the activities reported here, we mention some earlier systems that addressed issues of improving online textual communication for collaboration. These include CSILE and Knowledge Forum (for collaborative learning), HyperNews (a general threaded discussion forum tool), instant messaging systems such as AIM, and other chat tools. People have used these systems for work, education, and entertainment. Problem solving is included in these three areas, as some people solve problems as part of their jobs, as assignments in college classes, or for fun. Strengths and weaknesses of such systems, in terms of collaborative learning, are covered in a recent paper on
enriching the representational affordances for learners [4].

Here is a simple taxonomy for these systems. Although we focus on textual communication, other modalities such as images, sound clips, and videos can be embedded in textual material. We first distinguish two top-level categories of systems: message-style versus document-style. Within each of these, we have the orthogonal dimension of synchronous versus asynchronous. This results in four subcategories, the first of which is synchronous message-style textual communication systems. Chat and instant messaging populate this category. The next subcategory is asynchronous message-style systems, and these include email and threaded discussion forums. In the subcategory for synchronous document-style systems, we have wikis and shared documents such as those provided by Google Docs. In the asynchronous document-style subcategory, we have products such as Microsoft Word with its change-tracking and commenting features. With this taxonomy, we can more easily describe two of the systems that we used in the reported activities. One is the INFACT system, developed at the University of Washington. The other is Google Wave.

2.2. Specific Tools Used in Our Study

In our study we used these three tools: INFACT, Google Wave, and CoSolve. INFACT is an online, private-access threaded discussion forum with additional tools for educational assessment [5]. We used only its basic system, plus a feature called “the curtain” (a means to hide responses to a question, so participants can all respond before seeing the answers of others). INFACT’s forum falls squarely into the asynchronous message-style subcategory.

The second tool we used was Google Wave [2]. It provides group-editable online documents that are more interactive than the documents provided in Google Docs. For example, there is an explicit hierarchical structure to the contents of a “wave” and its components can be hidden or expanded like portions of a file system tree in, e.g., Windows Explorer. In addition, users edit the contents of a wave in a synchronous way that feels like typing in a chat session; others can see each character as it is typed. Thus it is like a wiki with a synchronous feel. A wave represents a shared document; however, users’ views of it may differ in terms of what is hidden or expanded. Although Wave falls in the document-style half of the taxonomy above, it extends into both the synchronous and asynchronous subcategories.

The third tool we used is an online service intended to support collaborative problem solving, called CoSolve. This tool is described in the next section.

2.3. Design methodology based on the classical theory of problem solving

Our design methodology is based on a desire to develop better support tools that will eventually facilitate teams of human designers and computer agents to effectively design a wide variety of artifacts more effectively than with existing tools. The methodology shares some of the goals of Meta-Design for end-user development [1]; however, ours is more restricted, because it is based on the classical theory of problem solving. This theory has its foundations in state-space-techniques for general problem solving and operations research [3]. State spaces are commonly taught in artificial intelligence courses as a means to automatically solve puzzles and play games. The use of state spaces in our present activity is at a ‘meta’ level, in which each state of the design space represents a set of restrictions on an eventual design (in our case, the design for a game), rather than as snapshot in the process of solving a puzzle. Our design space a very general one. In it, we apply operators that introduce axes called state variables or that specify their formal types. A state in this design space could be translated into a starting point for another state space exploration that adjusts the parameters identified in the first. Then a state in that parameter-adjusting space could be translated, with the help of a file describing the semantics of the game constructs, into a playable game (possibly mediated by computer).

A design step in this paper means adding a new state variable to an existing design state, or removing one, or changing its formal type. The formal type might be Boolean, or Enumeration over a set of n elements, for example. A design iteration consists of one or more design steps.

CoSolve is a web-based service built by our group to support collaborative problem solving online. It provides affordances for two kinds of users: problem solvers and problem posers. Solvers can work in teams to apply state-space operators to existing partial solutions to problems (or partial designs of objects), and they may solve a problem by creating a path to a goal state. Posers perform a kind of end-user-programming to create problem templates that solvers work with. A poser expresses a template by defining an initial state, a set of operators, and a method for graphical rendering of any state reachable in the template’s state space (which is completely specified by the initial state and the operators). We used CoSolve in the activity reported here to create a
concrete representation of two of the parameter-adjusting spaces. The problem-template facility hosted a set of operators for assigning values to the pre-programmed parameters for each of two classes of games.

3. Overall Design Process

Our design team consisted of ourselves: four graduate students in computer science and two faculty members, in architecture and in computer science, participating in a seminar on the study of problem solving and design, over the course of a 10-week academic term. The ideas for both games were developed within our group. Eco-avelli is a card game meant to model the political processes involved in the issue of global warming, and we used Google Wave and CoSolve for our communication. Go Atom is an elementary chemistry education game. For this game, we investigated using INFAC and CoSolve to support our design work.

In designing each of the games, we met face-to-face over several, roughly one-hour-long sessions, and we also used the online tools to communicate between sessions and during sessions. In each session, we conducted usually one or two design iterations. An iteration began by discussing the problems in the current version of the design, brainstorming and then deciding on a modified set of game rules, then playing the game, and analyzing how well the new rules enhanced or detracted from the game.

4. Design of Go Atom

![Go Atom game board with cards and atoms](image)

Figure 1. Example of a Go Atom game in progress. In the photo on the right, groups of atoms that form a functional group are circled by the player that won points for that group.

Go Atom is a game that involves basic chemistry concepts. A starting atom is placed at the center of the play area, and play consists of each player adding bonds and atoms from their own set of atoms to the center atom, in turn, to create a molecule, while following elementary chemistry bonding rules. Players gain points for each atom or bond they add to the structure, and the game ends when no more bonds or atoms can be added that follow these chemistry bonding rules. The player who ends the game gets to “claim” the molecule, and may receive points for it.

In designing this game, we wanted to explore how changing different aspects of the rules might affect the gameplay, while maintaining our two goals of maximizing the playability of the game (i.e., is it fun, enjoyable, easy-to-learn?), and teaching chemistry concepts. We hypothesized that by changing our scoring scheme, we would be able to come up with the right gameplay dynamics to achieve this balance.

The design process began with initial game idea proposals, posted as messages to INFAC, from members of the team. There were nine such messages, in a single thread. Once we decided on the Go Atom game, 40 additional messages, under nine different threads, were posted to INFAC in the design of this game. We held two face-to-face design sessions total, where we played through four design iterations: Iteration 1 and Iteration 2A & 2B (simultaneous iterations involving half the group in each iteration) during Session 1, and Iteration 3 during Session 2.

Before the first face-to-face session (Session 1) a problem template was created in CoSolve to describe the state space of the game design. Here is an example of our first state space: (i) scoring scheme for adding each type of atom and each bond, (ii) scoring scheme for claiming completed molecules, including points for each atom type and each bond, (iii) specification of the number and types of atoms in the initial hand of each player, (iv) whether or not initial hand is random, (v) if random, minimum allocations of atoms in the initial hand.

We tested our game designs by playing the game on pen and paper. Figure 1 shows two photographs from Iteration 3. As seen in the photos, playing cards were used as a way to randomly pick the initial hand of atoms of each player. On the right, we can see a play number at the bottom of each atom; we used these to record and identify each move made by each player during our play testing.

Some iterations were devoted to fixing gameplay issues. For example, after Iteration 1, we found that the scoring scheme was too unbalanced, with the player who claimed the atom having an unfair advantage in points, so we added a new rule that gave more chances for anyone to accumulate points: if a player completed all the bonds possible on an atom that was on the board (“closing” an atom). This was a rule that could not be accounted for in the initial state space, so it required adding additional variables to the state space.
Another example of the type of design problems we worked on is seen in Iteration 3. In this iteration, we considered the goal of increasing the educational value of the game and brainstormed ideas for doing so. One team member came up with the idea of a more Scrabble-like version of Go Atom, where points were awarded for building molecules that were actual chemical compounds that existed in nature, rather than simply any that fit the basic bonding rules. We tried this idea by adding a small “dictionary” of functional groups from organic chemistry. Players who built and identified functional groups within the molecule could claim them and win extra points. We played this version of the game, and found that this did increase our ability to identify functional groups, so we further discussed ways a more general “dictionary” of molecules could be developed for the game.

Table 1 shows the number of messages posted to INFACT during each stage of the design. Section 7 includes more detail on the nature of the messages posted to INFACT.

<table>
<thead>
<tr>
<th>Time</th>
<th>*</th>
<th>S1</th>
<th>*</th>
<th>S2</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Msgs</td>
<td>19</td>
<td>3</td>
<td>5</td>
<td>13</td>
</tr>
</tbody>
</table>

5. Design of Eco-avelli

The name “Eco-avelli” is a reference to how, in this game, Machiavellian techniques are at odds with sacrifices made for the greater good. We played the game with a regular deck of playing cards, and is meant to simulate the kinds of trade-offs nations must make in deciding how they will approach the issue of climate change.

Each player has a certain number of hidden as well as open cards in his or her hand, which are that player’s “resources.” Players take turns drawing cards from the deck and adding them to their hands. The more matching cards (combos) they have, the higher their points, and a scoring scheme specified the number of points single cards, pairs, three-of-a-kind, four-of-a-kind cards would generate, simulating the concept of countries being more powerful if they have the monopoly on a resource. However, on each turn, players are also allowed to request specific cards from each other, mimicking trade between countries.

Finally, there are also “convention” rounds, in which players vote to “ban” certain ranks of cards from being able to accumulate more points. The purpose of these convention rounds are to simulate climate change conventions where individual countries within the international community must negotiate what restrictions to put into place to limit carbon emissions for the greater good, while keeping their own country’s economic interests in mind.

In this design process, we again tried to balance the two goals of playability, with faithfulness to the educational aspect of the game. There were five design sessions of Eco-avelli in total, with eight iterations of the game during those sessions, including the first game session that was played before the official start of this study (Iteration 0). In Sessions 1 through 3, there were two design rounds each (Iterations 1 through 6), and in Session 4, there was one (Iteration 7).

Here is an example of the state space variables used before beginning Iteration 1: (i) size of each player's initial hand of cards, (ii) scoring scheme for combos, (iii) number of visible/hidden cards in a player's hand, (iv) rules for banning: effect of a ban, (v) rules for banning: majority vote? secret or open voting? etc., (vi) how are cards transferred between players, if allowed at all?

In most of our iterations, instead of simply tweaking one of these variables, we again found that we often needed to add a whole new variable to the state space all together. For instance, after Iteration 1, we found that since a player could declare victory immediately after drawing, if other players had not had their turn that round, the game was unfair and ended too quickly. So we brainstormed a variety of new rules, among them, a new rule which states you cannot declare victory immediately after your turn; you must wait until everyone has had another round of play. This is a rule that was not accounted for in the original variables of the state space, and hence, a variable was created for it, the “declare victory after turn”, and the variable’s value is set to “false.” However, once we included this variable in our state space, we never changed its value again.

As with the Go Atom experience, almost all of our design iterations involved trying to make the game more playable. This involved trying to implement rules to make the game fairer to all the players, to manage the length of gameplay, and to encourage player interaction and excitement during the game. We also attempted all the while to keep our secondary goal of making the game educational in that it reflect the real-world political process of climate change negotiations. However, fewer iterations were spent on this secondary goal.

We used CoSolve and Google Wave as communication and recording tools. In CoSolve, we created an initial problem template that described the state space as we viewed it after playing the game in Iteration 0. After Iteration 4, we found that we had increased the state space considerably in our iterations.
of the game, and posted a revised list of state variables into Wave, and then translated this into a revised CoSolve problem template, which was considerably larger (see Figure 2).

We then attempted to record all our previous iterations using this new state space model. However, after doing so, we did not make use of this CoSolve template in later iterations, as we found our state space kept changing and it required us to keep modifying the initial problem template.

In Wave, all the communication on Eco-Avell was done within one “wave.” There were 38 messages (“blips”) total in the wave, with four of these having been edited by more than one person, and 107 total actions taken on the Wave (this includes actions like adding users to the wave, or deleting things from the wave, etc.). The breakdown of the Wave actions by time is as follows:

Table 2: Actions Taken in Google Wave, by Session

<table>
<thead>
<tr>
<th>Time</th>
<th>*</th>
<th>S1</th>
<th>*</th>
<th>S2</th>
<th>*</th>
<th>S3</th>
<th>*</th>
<th>S4</th>
<th>*</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Actions</td>
<td>16</td>
<td>1</td>
<td>17</td>
<td>17</td>
<td>8</td>
<td>11</td>
<td>4</td>
<td>26</td>
<td>7</td>
</tr>
</tbody>
</table>

Most of our communication in Wave was in the form of recording what had taken place, the ideas that had been discussed, or the design decisions that had been made. In session 4, team members made a greater effort to use Wave, and hence, more actions were taken during that time period than in the others.

6. State Space Analysis / Iterations

Throughout our design process, we used a state space representation to model the current state of our design and the space of all alternatives under consideration. These representations were made explicit using CoSolve and described implicitly with INFACT and Wave, allowing us to record and study the evolution of our designs. We use two simple metrics, described below, to capture basic information about the design spaces we explored and the nature of the changes we made in the design process. These metrics are based on the state space model, helping us understand the extent to which typical design decisions can be understood as state space explorations. The fact that they are simple has the disadvantage that the measurements seem crude, but this is offset by the advantage that it is easy to understand any artifacts they introduce.

6.1. Modeling the Design Process

Formally, our design process consisted of a set of iterations, \( I \), and a relation predecessor(\( \eta_1, \eta_2 \)), \( \eta_1 \in I \) that indicates when the iteration \( \eta_1 \) is the direct predecessor of iteration \( \eta_2 \). In all cases we considered, the predecessor relation induced a directed tree structure on the set of iterations.

Each iteration \( \eta \in I \) is specified by the set of state variables and their values at that iteration. Given a state variable \( v \), we denote the value of that state variable at iteration \( \eta \) by \( \eta(v) \). If \( v \) is not in the set of state variables for \( \eta \), define \( \eta(v) = \bot \). We will need to refer to the set of state variables used in an iteration, and denote this by \( \textit{vars}(\eta) \). In our application, each state variable was either boolean, integer with a fixed number of bits, or a finite enumeration of options. Thus each state variable \( v \) can be assigned a size \( s(v) \) which is the number of bits needed to specify the value of \( v \).

When moving from one iteration to another, the following \textit{elementary design operations} changes may be applied:

1. The value of a state variable may change.
2. A new state variable may be added.
3. An existing state variable may be removed.
4. An existing state variable may be refined (split into multiple, new state variables).
5. Multiple existing state variables may be combined into a new, single state variable.
6. The domain of an existing state variable may be changed.

Changes of type 1 can be explored using standard state space exploration, as CoSolve was designed to support. Changes of type 2-6 involve modifications of the underlying state space and must be explored in a broader framework.

6.2 Design Process Metrics

At each design iteration, we measure the size of the state space under consideration and the size of the design change from the previous iteration. In particular we have

\textbf{Definition 1.} The effective state space size, \( \text{Size}(\eta) \), of an iteration \( \eta \in I \) is the number of bits needed to specify the state of the design to the design team at that iteration. Formally

\[
\text{Size}(\eta) = \sum_{v \in \text{vars}(\eta)} s(v)
\]
We can also measure the size of the change in an iteration by counting the number of state variables that changed to produce this iteration:

**Definition 2.** At an iteration $\eta$ let

$$\Delta(\eta) = \{ v \mid \eta'(v) \neq \eta(v) \land \text{predecessor}(\eta', \eta) \}$$

denote the set of changes state variables. We will call $|\Delta(\eta)|$ the *iteration magnitude* at $\eta$. It is the number of state variables that have different values in $\eta$ than in its predecessors.

With this framework in place, it is straightforward to define other more sophisticated metrics of design evolution. We propose these two as a simple first glance of the design process.

**Example 1.** Consider a simplified version of the state space for Eco-Avellii consisting of the following state variables and values at iteration $\eta_1$:

<table>
<thead>
<tr>
<th>State Variable (size in bits)</th>
<th>Number of Players (3 bits)</th>
<th>Initial Deck Size (6 bits)</th>
<th>Initial Hand Size (3 bits)</th>
<th>Victory Points (5 bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>6</td>
<td>40</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

After playing the game with these parameters, the design team decided that the game was too easy to win quickly, meaning there was less chance to develop strategy or collusion. Two proposals were made to improve the game: (1) increase the number of points needed to win to 15, and (2) make a player wait until the beginning of their next turn to declare victory (giving other players a chance to raid them in the interim). It was also proposed that (3) the number of players is not relevant and should not be considered as a state variable.

Change (1) required that we change the value of the state variable “Victory Points”, change (2) required that we add a new boolean state variable, call it “Can Declare Victory After Turn”, to specify whether we allow players to claim victory immediately after accumulating points in their turn. Finally, change (3) required us to remove the state variable “Number of Players”. These changes define the next iteration, $\eta_2$:

<table>
<thead>
<tr>
<th>State Variable (size in bits)</th>
<th>Initial Deck Size (6 bits)</th>
<th>Initial Hand Size (3 bits)</th>
<th>Victory Points (5 bits)</th>
<th>Can Declare Victory After Turn (1 bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>40</td>
<td>5</td>
<td>15</td>
<td>False</td>
</tr>
</tbody>
</table>

With the given bit sizes of each state variable, we can compute that $\text{Size}(\eta_1) = 17$, and $\text{Size}(\eta_2) = 15$. Furthermore, we can compute the magnitude of iteration $\eta_2$, three state variables ("Number of Players", "Victory Points", and "Can Declare Victory After Turn") changed in value, so $|\Delta(\eta_2)| = 3$.

### 6.3 The Results

The effective state space size and iteration magnitude were computed for each design iteration of Go Atom and Eco-Avellii. The results can be seen in Figures 2-5. First we observe the evolution of the state space size for each of the games (Figure 2 & Figure 3).

We see that in both cases our state space grew at each iteration. In particular this means we never saw an iteration characterized entirely by Type 1 changes. Thus pure state space search would not have been effective at this phase of the design process -- a fact that became clear to us as we saw our CoSolve state...
space specifications become obsolete after a few minutes of discussion in each round. This suggests that early stage design will benefit more from a more general state space designer, based on the elementary design operations 1-6.

We also measured the magnitude of each iteration (Figure 4 & Figure 5). Note that the iteration magnitudes and the growth rate of the state space both seemed to shrink in the later iterations of Eco-avelli's design. The experience of the design team did not entirely confirm this stabilization: while team members felt that the process was converging on a good version of a game, transcripts of the design sessions show that there was an explicit decision to avoid major changes in the last iteration. So the convergence may be an effect of external time constraints as much as it is an internal feature of the design process.

We also note that the state space encoding of an iteration is not unique—variables may be given different names, enumerated variables may be factored into Cartesian products of smaller variables, and unused variables may be added or removed. While these factors will not affect the qualitative nature of the results, inconsistent state space design will exaggerate the importance of some changes while understating others. Because of this, consistent state space design is important for effective quantification of design evolution.

7. Analysis of Communication Tools Used

One of the goals of this study was to observe how we would use different communication tools during the game design process.

We primarily used INFACT as a planning and record-keeping tool. For example, at the outset of our study we used a curtained thread and asked everybody to post his or her goals for the project. The curtain prevented any bias from reading others’ responses so that we would get the most accurate feedback possible. Likewise, we used INFACT to decide on specific goals for each face-to-face meeting and to summarize the set of rules used for each session of Go-Atom and how we felt about that iteration after playing.

Wave, on the other hand, we used largely as a supplement to the face-to-face meetings. Wave provides a more document-like model, making it a natural medium in which to record notes over time. We used it to transcribe the discussion and results of game tests. Since the transcriber could post and edit during design sessions instead of needing to combine everything into a single post at the end, posts in Wave had a much higher level of detail than those in INFACT, such as including the different suggestions made while considering new rules for Eco-avelli.

Given a chance to design another game, we would use the tools differently. Wave allows users to edit others’ posts with the right permissions, so we might use it for collaborative documents that would have proven useful over the course the design process, such as a design document specifying the criteria for success for either game or a brainstorming list of game ideas that we could refer to when trying different variations of the game. As it was, we occasionally forgot good game mechanisms proposed in previous sessions or wandered away from the global warming metaphor in favor of a more fun game.

Another improvement would have been to use tagging. Organizing posts by thread and date as we did made it difficult to search for old information. Instead, if we were to do it again, we would tag posts according to content. For example, a game mechanism
suggestion designed to make the Eco-avelli "conferences" better resemble real-world summits might have the tags [date] [mechanism] [global warming metaphor]. This would allow us to later remember what happened at that session by searching for the date, or to retrieve the game mechanism suggestion made if we had identified a problem about the game that we didn't immediately know how to solve, or remember why this change made Eco-avelli a better model of global warming politics.

It would also have been useful to collect hidden-response surveys from each player after an iteration, either using a curtained INFACT thread or a Wave voting gadget, to get each person's unbiased observations about a certain set of game rules, something which is inconvenient in a face-to-face setting. This type of data would also serve as a quantitative measure of how game design is progressing: over time, do players feel that the game meets its stated goals better than before? If not, then the design state might need to be reset to that of an earlier iteration.

Finally, it would have helped us a great deal to be able to play our games online, where all the moves would be automatically transcribed for later analysis and players would not need to physically meet to play. A virtual card deck programmed with common options (show/hide cards, draw cards, mark cards with text, discard cards) would have worked fine for this purpose. Other games would require some programming to set up an online game system, but the overhead would be worth the ability to play test online.

8. Conclusion

In this study of collaborative design, we have reflected upon our experience in designing these two educational games, Go Atom and Eco-avelli, and made observations on both the design process as well as how online communication tools can support this process.

We found that the state-space search model, as provided by CoSolve, not only gave us a way to communicate and think about the design process explicitly, but we were able to quantify our design changes in a concrete way through analysis of the state space variables of a design. In this analysis, we found that in almost all iterations, the state space grew, suggesting that perhaps in this early stage of design we engaged in, a simple state space search is not enough; rather, the construction of the state space through elementary design operations such as adding new state variables plays a bigger role.

Our analysis of the use of the communications tools INFACT, CoSolve, and Google Wave provided us with plenty of new ideas for ways to support collaborative design. The synchronous nature of Wave encouraged us to record more of the details of our design sessions during the actual sessions themselves. Also, we suggest ways of using features in INFACT or Wave to support future collaborations, such as curtained surveys for each team member to submit their feedback on each iteration. These surveys would have the dual benefit of quantitatively measuring the quality of the game, while also keeping in mind what the goals of each iteration is by reminding team members of these goals after each iteration.

While this work in still in its preliminary stages, we hope that the state-space-search model and our observations on the use of communication tools will help spur the creation of more efficient tools for communication and collaborative design.

9. References


