

The challenge of designing scientific discovery games

Seth Cooper¹, Adrien Treuille³, Janos Barbero¹, Andrew Leaver-Fay⁴, Kathleen Tuite¹, Firas Khatib², Alex Cho Snyder¹, Michael Beenen¹, David Salesin^{1,5}, David Baker², Zoran Popović¹, and >57,000 Foldit players⁶

¹Center for Game Science
Department of Computer Science & Engineering
University of Washington

{scooper,jbarbero,ktuite,axchos,beenen34,zoran}@cs.washington.edu

³Department of Computer Science
Carnegie Mellon University
truelle@cs.cmu.edu

⁴Department of Biochemistry
University of North Carolina
aleaverfay@gmail.com

²Department of Biochemistry
University of Washington
{firas,dabaker}@u.washington.edu

⁵Adobe Systems
salesin@adobe.com

⁶Worldwide

ABSTRACT

Incorporating the individual and collective problem solving skills of non-experts into the scientific discovery process could potentially accelerate the advancement of science. This paper discusses the design process used for Foldit, a multiplayer online biochemistry game that presents players with computationally difficult protein folding problems in the form of puzzles, allowing ordinary players to gain expertise and help solve these problems. The principle challenge of designing such *scientific discovery games* is harnessing the enormous collective problem-solving potential of the game playing population, who have not been previously introduced to the specific problem, or, often, the entire scientific discipline. To address this challenge, we took an iterative approach to designing the game, incorporating feedback from players and biochemical experts alike. Feedback was gathered both before and after releasing the game, to create the rules, interactions, and visualizations in Foldit that maximize contributions from game players. We present several examples of how this approach guided the game's design, and allowed us to improve both the quality of the gameplay and the application of player problem-solving.

Categories and Subject Descriptors

D.2.2 [Software Engineering]: Design Tools and Techniques; K.8.0 [Personal Computing]: General—*games*

1. INTRODUCTION

Games have recently been used to aid science by leveraging human image-recognition abilities, for example, to locate celestial objects [11]. This paper introduces a more general class of *scientific discovery games* that focus on leveraging human problem solving ability to solve computationally difficult scientific problems. A scientific discovery game

translates a class of computationally difficult scientific problems into puzzles, and provides a game-like mechanism for non-expert players to help solve these problems. Many traditional aspects of game design apply to scientific discovery games, including the design of introductory levels to draw newcomers and explain game mechanics, the use of a client-server architecture for competition and collaboration, and the requirement that the game be fun. However, unlike games whose goal is entertainment or education, scientific discovery games introduce a unique challenge: *enabling non-expert natural problem solvers to advance a specific scientific domain*. This challenge influences all aspects of the game design. First, visualization and graphics need to promote human ability to see complex solutions and convey accurate scientific information while remaining accessible to beginners. Second, interaction design must optimize for natural interactions suitable for the human exploration process, while still respecting scientific constraints. Finally, the scoring mechanism needs to be informative enough to promote multiple human strategies, while remaining true to the latest models of the underlying scientific phenomenon. Perhaps the most distinguishing feature and the greatest difficulty of design for this type of game is that the solution to the scientific problem, and thus the solution to the corresponding puzzles, is unknown. Since we do not know the solution *a priori*, we cannot design the game with specific solutions in mind.

To explore this space, we focused on human ability to reason about 3D structures and on the biochemistry domain, where many problems tend to be structural. We developed *Foldit*, a biochemical discovery game. In this paper, we discuss Foldit's initial focus on protein structure prediction – determining a protein's shape given its sequence of constituent amino acids. Protein structure prediction involves finding favorable interactions that form when the protein's chemical groups come into contact – essentially a 3D jigsaw puzzle. We believe that humans' innate spatial reasoning ability makes it possible for non-experts to make useful contributions to this problem. We leverage expert knowledge to shape the rules of the game, thus enabling a much larger pool of non-experts to make discoveries within this framework. Over the first two years since Foldit's public release in May 2008, we have run roughly 600 structure prediction

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

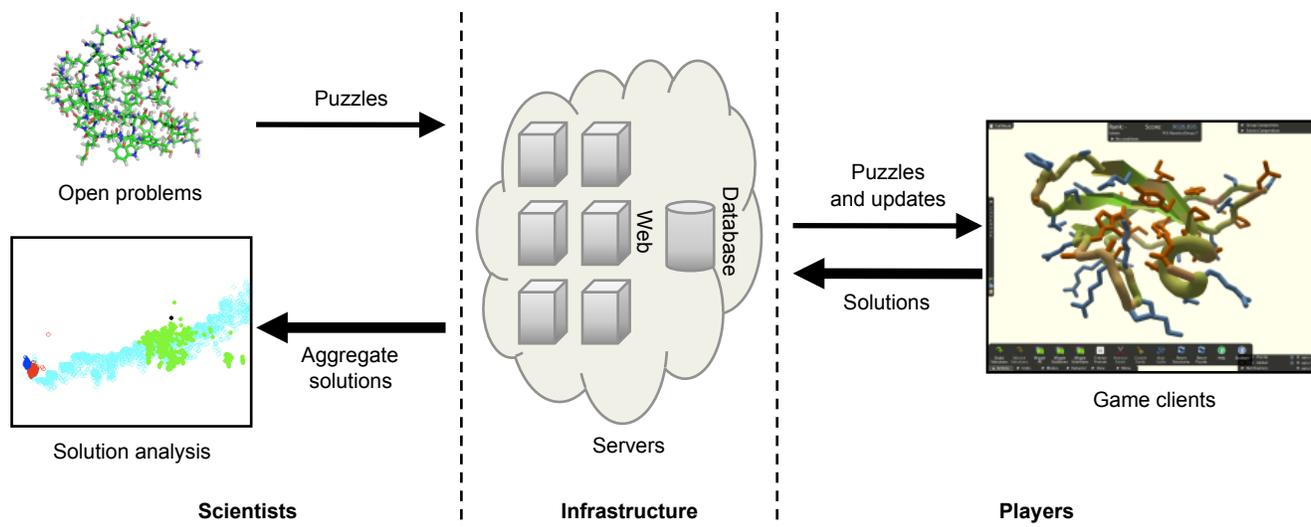


Figure 1: Overview of our architecture for scientific discovery games. The biochemistry team provides structure prediction and design problems for the server. These problems become puzzles and are sent to each player’s client. Players collaborate and compete to solve these problems and upload their solutions to the server, where they are aggregated and sent back to the biochemistry team for analysis. This analysis can then be used to improve the design of the game and puzzles.

puzzles and had over 57,000 players from a wide variety of backgrounds participate.

The rest of this paper describes our experience designing Foldit, with a special emphasis on the unique challenges posed by making biochemistry problems accessible to anyone. The creation of Foldit was a challenging and multidisciplinary project, drawing together computer science, art, game design and biochemistry. Moreover, we did not know ahead of time which parts of the problem players would be best at solving, or which in-game manipulation tools they would use most effectively. The only way to find out was to have people play Foldit. In order to deal with these and other uncertainties, we took an iterative approach both before and after releasing the game to the public. We have continually evolved the gameplay in response to massive gameplay traces, player feedback and expert analysis, and continue even now with this iterative process as we add features and expand the set of biochemical problems to which the Foldit community can contribute.

2. RELATED WORK

Games are often designed with an iterative approach, which involves designing, testing, and evaluating repeatedly until the player’s experience meets some criteria [10]. For most games, the main criterion for the player’s experience is simply to have fun. Player feedback and playtesting are an integral part of the process, and there are a number of methods of gathering and incorporating this information from players [1]. We have also continued the design process after the game’s release, to incorporate data gathered from the players in a continual process of evolutionary redesigning [12]. Our work differs from the standard iterative approach in that the game design space is constrained to conform with existing physical models, and we include the input of scientific experts in the evaluation of the game.

Recently, there has been much interest in using games as a means of motivating people to perform tasks that are currently difficult for computers. Games such as the ESP game [23] and Peekaboom [24] use human image-recognition ability to produce labeled images from gameplay. Image recognition has also been used for finding particular features of interest in scientific data, such as looking for signs of interstellar dust [25], measuring and aligning features on a planet’s surface [15], and classifying galaxy shapes [11]. Most such work is heavily image-based, and these projects have been successful in motivating players to sift through large image sets, which would otherwise be a mundane task. Some games have taken a slightly different approach, such as looking for solutions to graph based problems [7]. Our work is different because it leverages a deeper human problem solving ability to create novel scientific results.

More generally, all *serious games* have a purpose beyond entertainment that ranges from fitness and health [26], to training [4], to social change [22]. In our work, the main goal is to generate useful scientific discoveries; however, other aspects of game design, such as the requirement that the game be fun, contribute to achieving this goal, as the results rely on players playing the game.

There have been many purely computational approaches to protein structure prediction, including distributed computing projects such as Rosetta@home [21] (built on top of the BOINC distributed computing interface [2]). Atomic simulations of proteins have also been performed by distributed computing [9, 16].

3. OVERVIEW

3.1 Background

Predicting protein structures computationally is a central goal for computational biochemists because so much can be

understood about a protein’s function once its structure is known, and because it is so challenging to observe a protein’s structure directly. Proteins are central to biochemistry because they are the primary chemical for almost all cellular processes. DNA, a perhaps more widely recognized cellular chemical, derives its entire purpose in encoding protein sequences.

DNA encodes a protein by describing the linear *sequence* of amino acids that compose the protein. Cells translate a sequence of DNA into a sequence of amino acids, and then the resulting amino acid chain (the protein) folds into a unique, compact *structure* – often called the protein’s *native structure*. It is well known that sequence determines structure [3].

The native structure is one that is lowest in free energy – it has the most favorable set of chemical interactions. Some interactions involving the *backbone* – the repeating pattern of atoms that connects all the amino acids in the chain – occur so frequently that the structures they form have special names. These so called *secondary structures* include tightly wound *helices* and extended *sheets*. The remaining interactions involve amino acid *sidechains*, which stick out from the backbone and differentiate the various amino acids.

Foldit is built on top of the Rosetta molecular modeling suite which has proven useful at a wide variety of protein modeling tasks [20, 5, 17, 13]. The suite contains an energy function which captures the interaction energies between protein elements, as well as a set of structural optimization subroutines. For protein structure prediction, structures closer to the native structure will have a lower energy than structures further away from it. Foldit uses this state-of-the-art energy function to compute player’s scores, and also takes advantage of the optimization routines Rosetta makes available.

3.2 Architecture

Here we give an overview of the architecture of Foldit, which can be seen at a high level in Figure 1. Foldit generally uses a client-server architecture. Each user downloads and installs the client, which then communicates with a central server to send information about the local player and get information about other players.

Scientists post problems to the server; in the case of Foldit, these are protein structures for which the players are meant to find the native structures. An initial protein structure is associated with metadata such as a title and description, and parametrization such as which energy function terms to use. We call these *puzzles*, and they are posted on the server for a fixed amount of time (usually a week). While a puzzle is active, players can download it and interactively reshape the protein to try to achieve the best score. This often requires significant changes to the puzzle structures, which are given in various partially-folded states, and in some cases need to be completely refolded from a straight line. Players’ structures, or *solutions*, are reported back to the server, and players are ranked against other players who are playing the same puzzle. Players can form groups with which to share their solutions through the server, allowing them to work together to find even better solutions than they could working alone. When one player *shares* a solution by

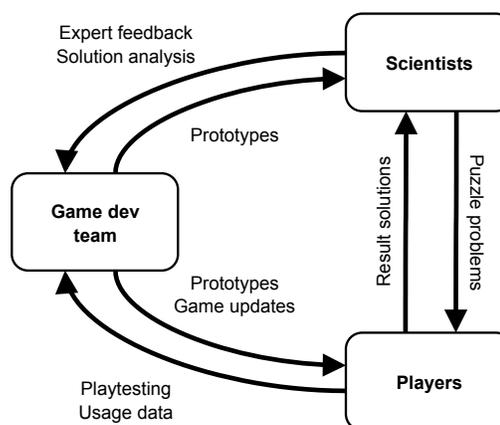


Figure 2: An overview of the interactions between the three groups.

uploading it to the server, other players in the same group are able to see it and download it. The social aspect of the game is supported by in-game chat, a website with forums, and a player-created wiki. At the close of a puzzle, the solution data is aggregated, and presented to the scientists for analysis.

The game is designed to be flexible, and the client allows automatic updating so that we can continually evolve the gameplay. The puzzle posting cycle and automatic updates allow us to respond to not only player feedback, but also to expert analysis, as we introduce and refine gameplay elements.

3.3 Iterative Strategy

In order to arrive at the current state of Foldit, we took an iterative approach to the game’s design. Given the complexity of this undertaking, we realized that it was unlikely that all our initial decisions would be the best. There are three major groups relevant to our approach: 1) the scientific experts whose problems the game is meant to help solve; 2) the players; and 3) the game development team. The development team must incorporate feedback from the players to make sure the game is understandable and fun, and from the experts to make sure that the results produced will be useful to them. An overview of the interactions between these three groups is given in Figure 2.

During the game’s initial development, the development team and experts must work together closely to determine an initial direction. This involves defining what problems to approach, what the fundamental gameplay mechanics needed are, and what the desired results are. Once possible games have been prototyped, player feedback can begin to be incorporated. Early playtesting helps to uncover what elements of the problem are fun and which can be most confusing and difficult to understand. This can help to both focus the gameplay and narrow the scope of the game to where players will most likely be able to contribute.

After making the game available to the public, a large amount of data and feedback can become available to help improve the game. As in a traditional game, data on gameplay can

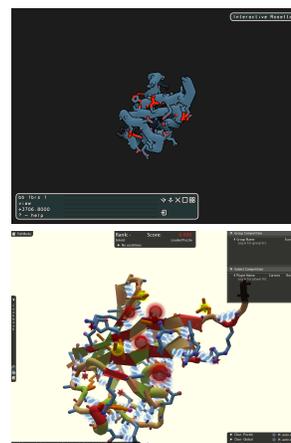
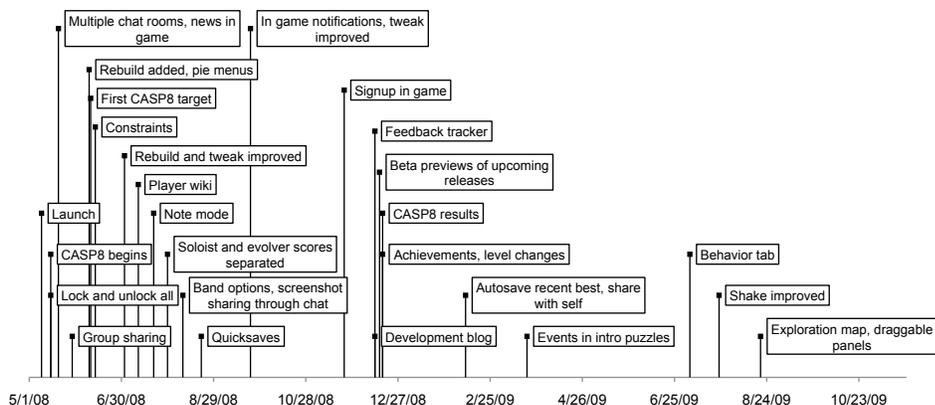


Figure 3: Selected events from the game’s evolution over time, with screenshots from before release (top) and the current version (bottom).

be gathered from players for an objective analysis of what players are doing, and feedback from the player community is extremely useful in determining new features. However, in a scientific discovery game, as scientists post puzzles and player solutions are analyzed, this analysis must then be incorporated in the design of the game, progressing towards ever better results.

Following this pattern, Foldit has evolved significantly since its initial release. A timeline of significant events in the evolution of the game are given in Figure 3.

4. DESIGN CHALLENGES

4.1 Visualizations

The visualizations in a scientific discovery game must achieve several purposes in order to allow players to apply their problem-solving skills. They must *reflect and illuminate the natural rules of the system*, in a way that makes state of the system evident to the player and directs them to where their contribution will be most useful. At the same time, the visualizations need to *manage and hide the complexity of the system*, so that players are not immediately overwhelmed by information. They must be *approachable by players* who have no knowledge of the scientific problem at hand. Thus, they should look inviting and fun, and not bring back memories of high school textbooks. Ideally, they should be customizable, because as with other aspects of the game, it is not clear from the outset what the best visualization will be, and different players may have different preferences.

In order to make the visualization of Foldit *reflect and illuminate* the fundamental properties of proteins, we worked with experts to distill simple rules upon which to base them. The first rule is to avoid clashes. Clashes occur when atoms are unrealistically close to each other, causing a large repulsive force. These can be prevented by keeping the atoms from overlapping, and are represented by spiky, rotating spheres that float between the overlapping atoms. The second rule is to fill voids, or empty spaces in the protein. Packing the protein tightly will remove voids. Voids are represented as bubble-like objects that pop when they come in contact with the protein. Clashes and voids appear red, as

natural proteins should not generally have any. The third rule is to bury exposed hydrophobics. Hydrophobics are sidechains whose chemical properties are such that it is favorable for them to be on the interior of the protein. Exposed hydrophobics are represented as small, pulsing spheres that move along their sidechain. These are drawn in yellow, rather than red, because natural proteins may have some exposed hydrophobics. The fourth rule is to maintain and create hydrogen bonds, which form between particular pairs of atoms and hold the protein together. Hydrogen bonds appear as undulating bars between the bonded atoms, and are drawn in blue, because they are good.

Due to the spatial nature of the problem, the visualization of the protein closely matches the actual geometry of the protein. To make the overall structure stand out, sheets, helices, and loops are stylized, similar to many expert visualization tools [8]. Sheets appear with a zig-zag pattern that will form hydrogen bonds when properly fit together. Color also plays a large role in the visualization of the protein. The backbone color reflects the score of the protein in a particular region – going from red in poor scoring regions to green in good scoring regions – so players can see where they can gain the most points. The sidechains are colored by hydrophobicity, so players can quickly see if they are extending them in the preferred direction. By coloring backbone and sidechain independently we can display more information while not introducing too much visual clutter.

Foldit takes a number of approaches to *manage and hide* the complexity of huge networks of interconnected atoms that make up a protein. Many unimportant details are hidden. Hydrogen atoms, which are plentiful on the protein but do not add a lot of structural information, are hidden. However, hidden information will reappear if it becomes important to the player: sidechains can disappear entirely to make the overall structure of the protein’s backbone clearer, but will reappear if they are causing a problem, such as if they are involved in a clash. Many actual clashes themselves are also hidden: only the worst clash is shown on a per-amino acid basis. This prevents the player from being overwhelmed by the number of clashes if the protein is compressed too tightly.

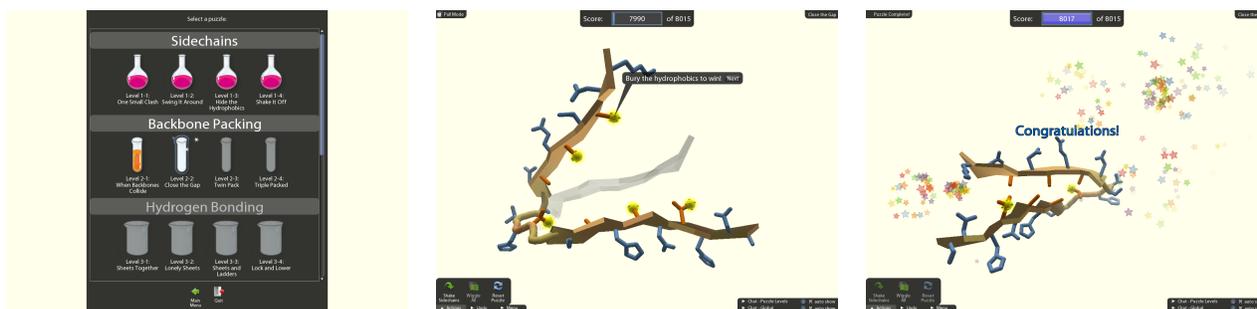


Figure 4: Flow through introductory levels. First, the player selects an available level from the menu. While playing the level, text bubbles pop up to guide the player. When the goal score is reached, a short reward animation is played.

To make the game *approachable*, we gave the protein itself a bright, cartoonish look. Many pieces of the visualizations move playfully around the protein. There are a wide variety of visualization options available in the game as well, such as alternative colorings and geometries for the protein. These can be accessed through a special menu option that is turned off by default. This approach allows more advanced players the ability to customize their view in the view options menu, but keeps things simple for newcomers.

Visualizations such as voids and exposed hydrophobics can be computationally expensive to compute. To keep the game interactive, we compute such visualizations in a separate thread, which will update the visualization after a delay.

4.2 Interactions

The interactions in a scientific discovery game must also achieve several purposes. They must *respect the constraints of the system* required. However, they must also be *sufficient to explore* the space of solutions enough to be able to solve the problem. They should also be as *intuitive and fun* as possible.

To ensure that the player interactions *respect the constraints* of protein folding, we developed a number of tools for players to use based on the powerful set of optimizations offered by Rosetta. By using Rosetta as a model for our interactions, we could ensure that they would result in plausible for protein structures. However, these optimizations only formed the basis for the moves, and they all needed to be adapted for interactive and intuitive use by players. The primary method of interaction in Foldit is directly manipulating the protein through pulling, by clicking and dragging the mouse. Depending on the location of the pull, this performs various Rosetta-based optimizations with the player’s pull as a soft constraint. There are also buttons to launch automatic algorithms for continuous energy minimization (*wiggle*); discrete sidechain energy optimization (*shake*); fragment insertion (*rebuild*); and the ability to rigidly rotate, translate, and shift sections of the protein (*tweak*). Players are able to achieve fine-grained control over these optimization through two methods. First, *freezing*, which prevents parts of the protein from moving, and second, *bands*, which can connect amino acids and pull on them independently of the player. When making large restructuring operations, the repulsion force between atoms can overpower what the player is trying to do and make it more difficult to interact

with the protein. To get around this, we have added a *behavior* menu, with a slider that allows player to adjust the strength of the repulsion during interactions.

There are additional modes for interaction that define what the mouse buttons do when the user clicks on the protein. The primary mode is the *pull mode*, described above. The *structure mode* allows the player to redefine the secondary structure labeling of the protein. This is done by directly assigning from a menu, or dragging existing labels across the protein. The *note mode* allows players to add their own notes and remarks to sections of the protein. These can be used by an individual or to communicate between players sharing solutions.

In order to confirm that the interactions available in Foldit were *sufficient to explore* enough protein structures to allow the players to make a discovery, we ran several puzzles in which the native structure was visible as a guide. With this native guide, players were able to use the tools in Foldit to get close to the native structure. The fact that players were able to do this suggested that the interactions would be sufficient to reach the native structure on unknown proteins as well.

Further, to encourage players to use the available interactions to explore the space in new ways, we added an *exploration map*. The exploration map plots every Foldit solution for a puzzle based on its score and how different it is from the puzzle’s starting structure. The map gives players a rough idea of the solution landscape: the different areas other players are exploring, and the scores they found there. Players might be exploring a new region on the map that initially gives a worse score, but by working hard in this new unexplored region, they might find a better shape and get the highest score.

In order to make the interactions more *intuitive and fun*, we followed the concept of *touchability* – being able to directly interact with the protein as though you could actually touch it. Before embracing this concept, our designs only manipulated the protein through indirect sliders, buttons, and plots. However, we soon changed the design to cause things to occur by clicking on the protein itself. While the major optimizations are still launched by buttons, actions like pulling, attaching bands, freezing, tweaking, and others are performed directly on the protein. This also led

us to the mode-based interface, which allowed us to use the mouse buttons for more operations by changing what they did in different modes.

The goals of interactivity and accuracy can sometimes conflict. The shake tool uses a discrete optimization over possible sidechain positions called rotamers. However, as the number of rotamers becomes large, running the optimization over all of them at once becomes too slow to be usable. Therefore, we rewrote the high level algorithm to run on small, spatially coherent sets of rotamers, rather than all at once. By running many shorter optimizations in sequence rather than one long one, players can see partial results and cancel the optimization early, while also getting the advantage of finer sampling from a larger number of rotamers.

Also, in order to allow local modifications to the backbone at interactive rates, we found it necessary to allow small non-idealities to occur in the protein structure – allowing degrees of freedom to vary that typically would remain fixed. A small amount of non-ideality was acceptable, so we allowed it, but subtracted a penalty from the score. We found that, initially, the level of non-ideality in the solutions was unacceptably high. After increasing the penalties, the non-idealities fell into a range that was acceptable to the experts, while still maintaining the desired player experience.

4.3 Scoring

By definition, the final outcome in a scientific discovery game is unknown. Therefore, we cannot base the goal of the game on reaching a particular known state. The goal must be one that will *direct players toward the solution* to the problem and encourages players to explore the space.

In Foldit, we want to motivate the players to find the best possible protein structures, yet we do not know what those structures are. To do this, we organize the game in the form of a competition, where a player’s goal is to do better than other players. In a sense, this allows players who find good structures to set the goal for the other players. To make sure that better scores will *direct players toward the solution*, we base scoring on the Rosetta energy function, which reports a lower energy for structures nearer the native. However, we negate the energy so players are competing for a higher score. The energy function is broken up into several terms – such as clashing or hydrogen bonding – based on where the contribution to the score is coming from, and this information is made available to the players.

In each puzzle, players are ranked by the score of the best solution they have found, and at the close of a puzzle, *global points* are assigned based on ranking. These global points are accumulated over time, allowing players to have an overall ranking against all other players. Groups of players are also ranked and scored in a similar fashion. Initially all players were ranked together in a single leaderboard. However, feedback from the players indicated that the solution sharing architecture was unfair to individuals working alone who had to compete against players in groups who could simply take another group member’s solution and move to the top of the rankings. To prevent this unfairness, we separated rankings into soloists, for players who worked without trading solutions, and evolvers, for players who worked by

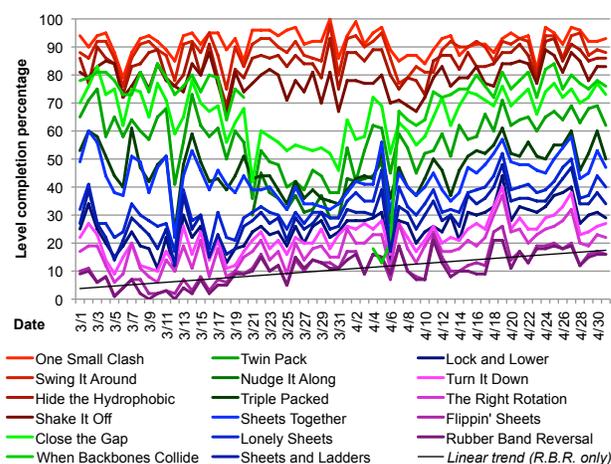


Figure 5: The completion percentage for each introductory level per day over a two month period. New players must start with the first level, ‘One Small Clash’, and have to successfully complete it before attempting the next level, ‘Swing it Around’, and so on until the last level: ‘Rubber Band Reversal’. On 03/21 all the intro levels were changed to use text bubble hints; the ‘Close the Gap’ level was removed and ‘Nudge it Along’ was added as a level. This closed the space between the blue levels, but created a large gap in the green levels. The removal of the level between the red and green lines seemed to be the main culprit, therefore on 04/06 ‘Close the Gap’ was restored and ‘Nudge it Along’ was removed. This fixed the large gap in the green levels while maintaining the overall shift upwards; the linear trend of new players completing all the intro levels increases over this two month period (grey line at the bottom).

improving other players’ solutions. This dealt with the issue of unfairness while allowing us to reward specialization. However, it did cause players to have to divide their efforts if they were interested in ranking highly in both.

4.4 Introductory Levels

We do not expect players to have a background in the scientific problem, or even be familiar with it. Thus, if players are to be successful, it is necessary to *teach players the system* and how the gameplay, visualizations, interactions, and scoring work.

In Foldit, we *teach* gameplay concepts through a series of introductory levels – offline puzzles that have an associated goal score, which, when reached, will complete the level, unlocking the next one. Each level introduces the player to new problem related concepts as though they are the rules of a game. These concepts are introduced through an event-driven system of “text-bubble” hints, where short text hints appear to guide the player based on what they are doing. The levels are designed such that using the newly introduced concepts is the easiest way to reach the goal score. The levels are organized into sets, each dealing with a particular high-level concept, such as hydrogen bonding. An example of the flow through an of an introductory level is shown in Figure

We refined these levels using both qualitative feedback from playtesting and quantitative data gathered from gameplay. First, before releasing levels, we performed think-alouds, where we would invite players to play through the levels and say out loud what they were thinking [19]; this would help us to determine where players were confused and what worked well, as well as what adjustments could be made to the gameplay to be more fun. In each round, we interviewed 3-5 people and collected the top complaints and points of confusion. Speaking with additional people each round would have led to diminishing returns where players would repeat the issues found by other players. After each round, we quickly made improvements to the game and ran another set of think-alouds, until the major issues were resolved.

Second, once the levels were released, we gathered data on how far players progressed through them. This gave us a good high-level view of where the most troublesome places were for most people. We could then focus our efforts on levels that caused the largest drop-off in players, make adjustments, and observe the results. An example of this process is given in Figure 5. We are able to aggregate data over a period of time, instead of looking at just a single day.

New players are not required to finish any introductory levels before trying out the online puzzles. To help bridge the gap between the levels and the online puzzles and keep players from being completely overwhelmed by a potentially complicated or difficult puzzle that has recently been posted, we include beginner online puzzles. These are similar to the other online puzzles, but available only to new Foldit players, who have fewer than 150 global points. Beginner puzzles are posted for one month and involve known solved proteins whose native fold has been randomly modified and is also shown as a transparent guide superimposed with the current solution. This native guide serves to give newer players an attainable goal, as well as an idea of what native protein folds look like.

5. EVALUATION

To evaluate if players could use the current design of Foldit to find the solutions to unknown scientific problems, we made several entries into the CASP8 competition in 2008 [14]. CASP is a semi-annual competition of protein structure prediction methods [6]. Sequences (called targets) whose structures are unpublished, but have or will soon be experimentally determined, are posted, and teams make their predictions of the native structure. This ensures that predictions are blind - no one entering the competition knows the solution to the problem. In CASP8, we participated primarily in *homology modeling* targets. The sequences of these targets (with unknown structure) are similar to sequences with known structures, and homology information from related structures can be useful for making predictions.

To prepare for CASP8, we ran several puzzles using homology targets from CASP7 in order to refine our methods. Because these targets were from the previous CASP, we had access to the native structures for evaluation. In the first CASP7 puzzles we ran, we found that the play-

Table 1: Rankings of predictions in CASP8 from the DBAKER team for targets which included Foldit solutions. Rankings for the remaining DBAKER prediction, which did not use Foldit, are also given. Rankings were based on the GDT_TS metric[27].

Target	Foldit rankings	Other DBAKER team rankings	Total entry count (all teams)
TR389	3, 7, 21	4, 18	71
TR432	14, 37, 48	1, 8	83
TR453	23, 46, 69	28, 50	85
TR461	2, 12, 19	1, 26	83
TR469	2, 3, 45, 50	4	74
TR488	1, 3, 26	2, 18	77
TS423	12, 42, 43	2, 31	496
TS492	2, 96	1, 3, 448	527
TS499	59, 107	33, 455, 504	519

ers would often find solutions further from the native than the puzzle’s starting structure. Moving away from the native is typical problem in homology modeling when refining structures. Consulting with biochemists, we decided to add expert-defined penalty constraints to the scoring of the puzzles. These constraints integrated homology information to prevent the structures from moving in less plausible ways. After running puzzles with these constraints, we found that players could improve on the quality of the starting structures.

Foldit player solutions were submitted for nine different CASP8 targets as part of the DBAKER team. For each target, several puzzles were run with various parameters and starting structures derived from homology information, in order to give the players a variety of places to start from in their search. The solutions from these puzzles, often numbering in the thousands, were aggregated and presented to the experts. From this pool, the experts selected submissions by a process of clustering the lowest energy structures, then selecting from those based on cluster size, energy, and visual inspection. This is similar to the process used to select automated prediction submissions from a large pool [18].

The resulting rankings are given in Table 1, for both the submissions that came from Foldit player solutions and the remaining DBAKER submissions that did not involve Foldit. It is worth pointing out that this ranking (by the GDT_TS metric [27]) is just one of many possible for CASP assessment; for all but one of these targets the majority of prediction from the DBAKER team involved Foldit, and the puzzle constraints placed significant limits on what the players were able to do. However, we believe these results support the conclusion that the game has been designed in such a way that players can use it to solve scientific problems.

6. CONCLUSION

Unlike most video games, where entertainment is the main goal and the design is entirely up to the creators, the design of Foldit was guided primarily by enabling anyone with a PC to take part in scientific problem solving. One of the most difficult aspects of development was that we were designing a game in which the final outcome was not known. Even

in Foldit puzzles where the best structure is unknown, the game has to guide the player toward that structure.

We have described how we overcame these unique challenges and the iterative approach we took to designing the Foldit. We applied this approach to the visualizations and interactions in the game, as well as introducing the necessary concepts to players.

In designing Foldit, we have learned the importance of including iterative adjustments to the game in the process of design, as the initial decisions can always be improved upon. We have also learned not to expect the way that expert scientists view the problem to be the best way for players. Exploring different avenues for looking at scientific problems can lead to new and useful opportunities for problem solving. Thus, we have also found that it is possible to make a fun experience by focusing on the exciting aspects of scientific problems, as opposed to the textbook details. We can take lessons from traditional game design to do this: rewarding players and keeping them interested are necessary for any game.

We plan to continue improving Foldit and applying this approach to allow discoveries in biochemistry and even more scientific domains. Further, we believe this approach can be applied to other spatial reasoning problems, allowing players to contribute to advancing the frontiers of knowledge.

7. ACKNOWLEDGMENTS

The authors would like to thank Joshua Snyder, Daniel Suskin, Philipp Krähenbühl, Hao Lü, Lu Sien Tan, Jeehyung Lee, Alex Chia, Ming Yao, Eric Butler, Chris Carrico, Philip Bradley, Ian Davis, David Kim, Rhiju Das, Will Sheffler, James Thompson, David Anderson, Yutong Zhao, Sayer Herin, and Benjamin Bethurum for their help. We would also like to acknowledge all the Foldit players who have made this work possible. This work was supported by NSF grants IIS0811902 and 0906026, DARPA grant N00173-08-1-G025, the DARPA PDP program, the Howard Hughes Medical Institute (D.B.), Microsoft, and an NVIDIA fellowship. This material is based upon work supported by the National Science Foundation under a grant awarded in 2009.

8. REFERENCES

- [1] M. Ambinder. Valve's approach to playtesting: The application of empiricism. Game Developer's Conference, Mar. 2009.
- [2] D. P. Anderson. BOINC: A system for public-resource computing and storage. In *5th IEEE/ACM International Workshop on Grid Computing.*, 2004.
- [3] C. B. Anfinsen. Principles that govern the folding of protein chains. *Science*, (181):223–230, 1973.
- [4] P. Backlund, H. Engström, C. Hammar, M. Johannesson, and M. Lebram. Sidh – a game based firefighter training simulation. In *11th International Conference Information Visualization (IV '07)*, pages 899–907, July 2007.
- [5] P. Bradley, K. M. S. Misura, and D. Baker. Toward high-resolution de novo structure prediction for small proteins. *Science*, 309(5742):1868–1871, September 2005.
- [6] CASP. <http://predictioncenter.org/>.
- [7] C. Cusack, C. Martens, and P. Mutreja. Volunteer computing using casual games. In *Proceedings of Future Play 2006 International Conference on the Future of Game Design and Technology*, 2006.
- [8] W. L. DeLano. The pymol molecular graphics system, 2002.
- [9] Folding@home. <http://folding.stanford.edu/>.
- [10] T. Fullerton. *Game Design Workshop: A Playcentric Approach to Creating Innovative Games*. Morgan Kaufmann, 1 edition, April 2008.
- [11] Galaxy Zoo. <http://www.galaxyzoo.org/>.
- [12] D. Kennerly. Better game design through data mining. *Gamasutra*, August 2003.
- [13] B. Kuhlman, G. Dantas, G. Ireton, G. Varani, B. Stoddard, and D. Baker. Design of a novel globular protein fold with atomic-level accuracy. *Science*, 302:1364–1368, 2003.
- [14] J. Moulton, K. Fidelis, A. Kryshchuk, B. Rost, and A. Tramontano. Critical assessment of methods of protein structure prediction – round VIII. *Proteins: Structure, Function, and Bioinformatics*, 77(S9):1–4, 2009.
- [15] NASA Be a Martian. <http://beamartian.jpl.nasa.gov/>.
- [16] V. S. Pande, I. Baker, J. Chapman, S. P. Elmer, S. Khaliq, S. M. Larson, Y. M. in Rhee, M. R. Shirts, C. D. Snow, E. J. Sorin, and B. Zagrovic. Atomistic protein folding simulations on the submillisecond time scale using worldwide distributed computing. *Biopolymers*, 68:91–109, 2003.
- [17] B. Qian, S. Raman, R. Das, P. Bradley, A. J. McCoy, R. J. Read, and D. Baker. High-resolution structure prediction and the crystallographic phase problem. *Nature*, 450(7167):259–264, November 2007.
- [18] S. Raman, R. Vernon, J. Thompson, M. Tyka, R. Sadreyev, J. Pei, D. Kim, E. Kellogg, F. DiMaio, O. Lange, L. Kinch, W. Sheffler, B.-H. Kim, R. Das, N. V. Grishin, and D. Baker. Structure prediction for casp8 with all-atom refinement using rosetta. *Proteins: Structure, Function, and Bioinformatics*, 77(S9):89–99, July 2009.
- [19] J. Ramey, T. Boren, E. Cuddihy, J. Dumas, Z. Guan, M. J. van den Haak, and M. D. T. De Jong. Does think aloud work? How do we know? In *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*, pages 45–48, New York, NY, USA, 2006. ACM.
- [20] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker. Protein structure prediction using Rosetta. *Methods Enzymol*, 383:66–93, 2004.
- [21] Rosetta@home. <http://boinc.bakerlab.org/rosetta/>.
- [22] K. Schreiner. Digital games target social change. *IEEE Computer Graphics and Applications*, 28(1), Jan./Feb. 2008.
- [23] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326, New York, NY, USA, 2004. ACM.
- [24] L. von Ahn, R. Liu, and M. Blum. Peekaboom: a game for locating objects in images. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64, New York, NY, USA, 2006. ACM Press.
- [25] A. J. Westphal et al. Non-destructive search for interstellar dust using synchrotron microprobes. In *Proceedings of ICXOM20, 20th International Congress on X-ray Optics Microanalysis*, 2009.
- [26] Wii Fit. <http://www.nintendo.com/wiifit/>.
- [27] A. Zemla. LGA: A method for finding 3D similarities in protein structures. *Nucleic Acids Res*, 31(13):3370–3374, July 2003.