Data generated as a side effect of game play also solves computational problems and trains AI algorithms.

BY LUIS VON AHN AND LAURA DABBISH

Designing Games With A Purpose

MANY TASKS ARE trivial for humans but continue to challenge even the most sophisticated computer programs. Traditional computational approaches to solving such problems focus on improving artificial-intelligence algorithms. Here, we advocate a different approach: the constructive channeling of human brainpower through computer games. Toward this goal, we present general design principles for the development and evaluation of a class of games we call "games with a purpose," or GWAPs, in which people, as a side effect of playing, perform tasks computers are unable to perform.

The Entertainment Software Association (www.theesa.com/facts/gamer_data.php) has reported that more than 200 million hours are spent each day playing computer and video games in the U.S. Indeed, by age 21, the average American has spent more than 10,000 hours playing such games[15]—equivalent to five years of working a full-time job 40 hours per week.

cute cute red animal nose

whale ocean ocean water

fish kid trees rock rock

scenery world world dark dark

branch branch

race red purple blue blue blue tire derby green

picture picture smile

north pole ocean lake snow snow

boy teeth you woman kid green green point girl

man music

sea stone rock rock

war sand guns yellow gun gun

movie johnny depp fire fire

clouds house green green

black white cold cold

photo ice picture standing cover

Beatles FOR SALE

Channel Islands National Marine Sanctuary

Shane Anderson

MAINLANDS PLAZA · CALVARY CHAPEL

Why I love my little black dress

Proof... not promises:
Introducing a scientific breakthrough
prevage anti-aging treatment

A. Einstein 40
1879-1955 HELVETIA

The NEW WORLD

What if this time and energy were also channeled toward solving computational problems and training AI algorithms?

People playing GWAPs[22–25] perform basic tasks that cannot be automated. The ESP Game,[22] a.k.a. the Google Image Labeler (images.google.com/imagelabeler/), is a GWAP in which people provide meaningful, accurate labels for images on the Web as a side effect of playing the game; for example, an image of a man and a dog is labeled "dog," "man," and "pet." The game is fast-paced, enjoyable, and competitive; as of July 2008, 200,000 players had contributed more than 50 million labels; try it yourself at www.gwap.com. These labels can be used to improve Web-based image search, which typically involves noisy information (such as filenames and adjacent text). Rather than using computer-vision techniques that do not work well enough, the ESP Game constructively channels its players to do the work of labeling images in a form of entertainment.

Other GWAPs include Peekaboom,[25] which locates objects within images (and has been played more than 500,000 human-hours); Phetch,[23] which annotates images with descriptive paragraphs; and Verbosity,[24] which collects commonsense facts in order to train reasoning algorithms. In each, people play not because they are personally interested in solving an instance of a computational problem but because they wish to be entertained.

The ESP Game, introduced in 2003, and its successors represent the first seamless integration of game play and computation. How can this approach be generalized? Our experience building and testing GWAPs with hundreds of thousands of players has helped us spell out general guidelines for GWAP development. Here, we articulate three GWAP game "templates" representing three general classes of games containing all the GWAPs we've created to date. They can be applied to any computational problem to construct a game that encourages players to solve problem instances. Each template defines the basic rules and winning conditions of a game in a way that is in the players' best interest to perform the intended computation. We also describe a set of design principles that comple-

ment the basic game templates. While each template specifies the fundamental structure for a class of games, the general design principles make the games more enjoyable while improving the quality of the output produced by players. Finally, we propose a set of metrics defining GWAP success in terms of maximizing the utility obtained per human-hour spent playing the game.

Related Work

Though previous research recognized the utility of human cycles and the motivational power of gamelike interfaces, none successfully combined these concepts into a general method for harnessing human processing skills through computer games.

Networked individuals accomplishing work. Some of the earliest examples of networked individuals accomplishing work online, dating to the 1960s, were open-source software-development projects. These efforts typically involved contributions from hundreds, if not thousands, of programmers worldwide. More recent examples of networked distributed collaboration include Wikipedia, by some measures equal in quality to the *Encyclopaedia Britannica*.[6]

The collaborative effort by large numbers of networked individuals makes it possible to accomplish tasks that would be much more difficult, time consuming, and in some cases nearly impossible for a lone person or for a small group of individuals to do alone. An example is the recent Amazon Mechanical Turk system (developed in 2005, www.mturk.com/mturk/welcome) in which large computational tasks are split into smaller chunks and divvied up among people willing to complete small amounts of work for some minimal amount of money.

Open Mind Initiative. The Open Mind Initiative[18,19] is a worldwide research endeavor developing "intelligent" software by leveraging human skills to train computers. It collects information from regular Internet users, or Netizens, and feeds it to machine-learning algorithms. Volunteers participate by providing answers to questions computers cannot answer (such as "What is in this image?"), aiming to teach computer programs common-

> **People play not because they are personally interested in solving an instance of a computational problem but because they wish to be entertained.**

sense facts. However, the Open Mind approach involves two drawbacks: reliance on the willingness of unpaid volunteers to donate their time and no guarantee that the information they enter is correct. GWAPs differ from Open Mind in that they are designed to be enjoyable while ensuring that the data they collect is free from error.

Interactive machine learning. Another area leveraging human abilities to train computers is "interactive machine learning"[4] in which a user provides examples to a machine-learning system and is given real-time feedback as to how well an algorithm is learning. Based on the feedback, the user is able to determine what new examples should be given to the program. Some instances of this approach have utilized human perceptual skills to train computer-vision algorithms to recognize specific objects.

Making work fun. Over the past 30 years, human-computer-interaction researchers have recognized and written about the importance of enjoyment and fun in user interfaces.[16,26] For example, systems (such as the StyleCam) aim to use gamelike interaction to increase enjoyment and engagement with the software.[21] Many researchers have suggested that incorporating gamelike elements into user interfaces could increase user motivation and the playfulness of work activities.[16,26] Some projects have taken this notion further, turning the user interface itself into a game. For instance, PSDoom provides a first-person-shooter-style interface for system-administrator-related tasks.[2,3] The idea of turning work tasks into games is increasingly being applied in children's learning activities.[12] Researchers note, as we do here, that it is important to not simply slap a gamelike interface onto work activities but to integrate the required activities into the game itself; there must be tight interplay between the game interaction and the work to be accomplished.

Desire to Be Entertained
The GWAP approach is characterized by three motivating factors: an increasing proportion of the world's population has access to the Internet; certain tasks are impossible for computers but easy for humans; and people spend lots of time playing games on computers.

In contrast to other work that has attempted to use distributed collections of individuals to perform tasks, the paradigm we describe here does not rely on altruism or financial incentives to entice people to perform certain actions; rather, they rely on the human desire to be entertained. A GWAP, then, is a game in which the players perform a useful computation as a side effect of enjoyable game play. Every GWAP should be associated with a computational problem and therefore generate an input-output behavior.

A game can be fully specified through a goal players try to achieve (the winning condition) and a set of rules that determines what players can and cannot do during the game. A GWAP's rules should encourage players to correctly perform the necessary steps to solve the computational problem and, if possible, involve a probabilistic guarantee that the game's output is correct, even if the players do not want it to be correct.

The key property of games is that people want to play them. We therefore sidestep any philosophical discussions about "fun" and "enjoyable," defining a game as "successful" if enough human-hours are spent playing it.
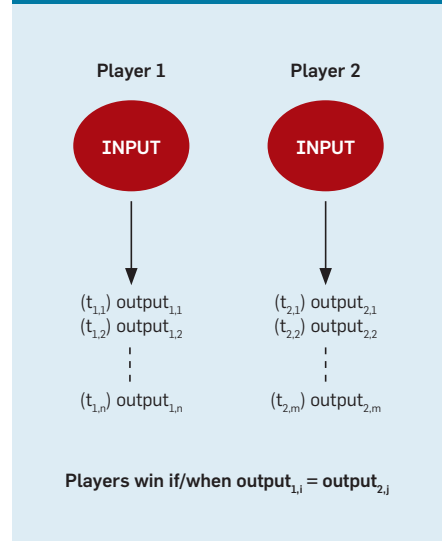
We advocate a transformative process whereby a problem is turned into a GWAP. Given a problem that is easy for humans but difficult or impossible for computers, the process of turning the problem into a GWAP consists of first creating a game so that its structure (such as rules and winning condition) encourages computation and correctness of the output. Having created many GWAPs, including the ESP Game, Peekaboom, Phetch, and Verbosity, we explore three game-structure templates that generalize successful instances of human computation games: output-agreement games, inversion-problem games, and input-agreement games.

Output-agreement games. Output-agreement games (see Figure 1) are a generalization of the ESP Game (see the sidebar "The ESP Game and Verbosity" on page 65) to its fundamental input-output behavior:
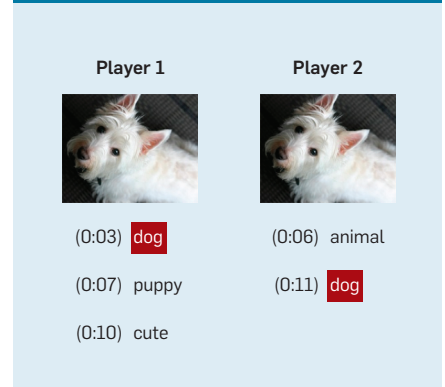
Initial setup. Two strangers are randomly chosen by the game itself from among all potential players;

Rules. In each round, both are given the same input and must produce out-

Figure 1: In this output-agreement game, players are given the same input and must agree on an appropriate output.

Player 1   Player 2

INPUT   INPUT

$(t_{1,1})$ output$_{1,1}$ $(t_{2,1})$ output$_{2,1}$
$(t_{1,2})$ output$_{1,2}$ $(t_{2,2})$ output$_{2,2}$

$(t_{1,n})$ output$_{1,n}$ $(t_{2,m})$ output$_{2,m}$

Players win if/when output$_{1,i}$ = output$_{2,j}$

Figure 2: In this output-agreement game, the partners are agreeing on a label.

Player 1   Player 2

(0:03) dog   (0:06) animal
(0:07) puppy   (0:11) dog
(0:10) cute

puts based on the input. Game instructions indicate that players should try to produce the same output as their partners. Players cannot see one another's outputs or communicate with one another; and
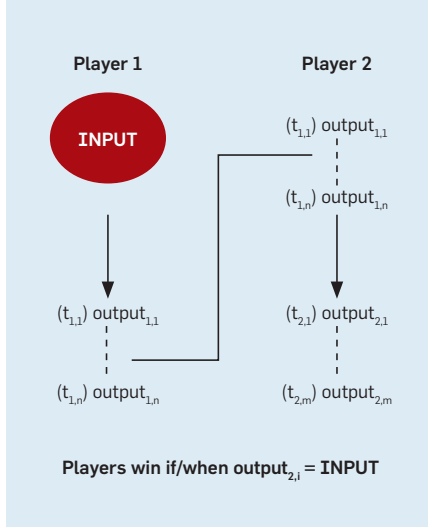
Winning condition. Both players must produce the same output; they do not have to produce it at the same time but must produce it at some point while the input is displayed onscreen.

When the input is an image and the outputs are keyword descriptions of the image, this template becomes the ESP Game (see Figure 2).
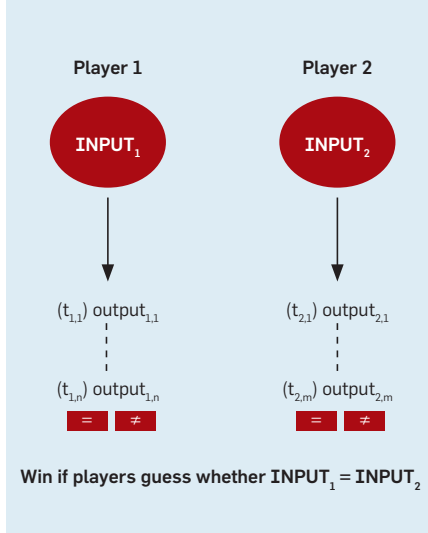
Since the two players cannot communicate and know nothing about each other, the easiest way for both to produce the same output is by entering something related to the common input. Note, however, that the game rules do not directly tell the players to enter a correct output for the given

Figure 3: In this inversion-problem game, given an input, Player 1 produces an output, and Player 2 guesses the input.

Player 1   Player 2

INPUT

$(t_{1,1})$ output$_{1,1}$

$(t_{1,n})$ output$_{1,n}$

$(t_{1,1})$ output$_{1,1}$ $(t_{2,1})$ output$_{2,1}$

$(t_{1,n})$ output$_{1,n}$ $(t_{2,m})$ output$_{2,m}$

Players win if/when output$_{2,i}$ = INPUT

Figure 4: In this input-agreement game, players must determine whether they have been given the same input.

Player 1   Player 2

INPUT$_1$ INPUT$_2$

$(t_{1,1})$ output$_{1,1}$ $(t_{2,1})$ output$_{2,1}$

$(t_{1,n})$ output$_{1,n}$ $(t_{2,m})$ output$_{2,m}$

=   ≠   =   ≠

Win if players guess whether INPUT$_1$ = INPUT$_2$

input; all they know is that they must "think like each other" and enter the same output.

This game structure accomplishes several goals at once: a good "winning" strategy for the players is to produce outputs related to the only thing they have in common—the input; when the two players provide the same output, this partially verifies that the output is correct, since it comes from two largely independent sources; and trying to agree on the same output with a partner is an enjoyable social experience.

Inversion-problem games. Resulting from any of three seemingly different games—Peekaboom,[25] Phetch,[23]

and Verbosity[24]—they, in their most general form (see Figure 3), can be described through the following rules:

Initial setup. Two strangers are randomly chosen by the game itself from among all potential players;

Rules. In each round, one player is assigned to be the "describer," and the other player is assigned to be the "guesser." The describer is given an input. Based on this input, the describer produces outputs that are sent to the guesser. The outputs from the describer should help the guesser produce the original input; and

Winning condition. The guesser produces the input that was originally given to the describer.

Verbosity (see the sidebar) is an inversion-problem game where the input is a word and the outputs are commonsense facts related to that word. Following the input word "milk," the game might output such facts as "it is white" and "people usually eat cereal with it."

Although the design of Verbosity involves other game elements not described here, the basic idea is that players need not be asked directly for facts about "milk." The game is designed such that facts are collected as a side effect of playing. Players told to "please enter facts about milk" might not be motivated to do so or enter incorrect information.

In inversion-problem games, partners are successful only when the describer provides enough outputs for the guesser to guess the original input. If the outputs are incorrect or incomplete, the guesser will not be able to produce the original input. Therefore, the game structure encourages players to enter correct information. At the same time, having one player guess the input while the other describes it is an enjoyable social interaction, similar to the popular children's game "20 Questions."

Additional elements can be added to inversion-problem games to increase player enjoyment, including transparency and alternation:

Transparency. In post-game questionnaires, players of inversion-problem games have expressed a strong desire to see their partner's guesses. We therefore experimented with adding a level of transparency between players

so the actions of one would be visible to the other. In games like Verbosity and Peekaboom this transparency is achieved by displaying partner guesses to the describers and allowing them to indicate whether each guess is "hot" or "cold." This design feature increases the social connection between the players without compromising output correctness.

Alternation. Unlike output-agreement games (where both players continually perform the same task), inversion-problem games are asymmetric in that each player in the pair performs a different task. In some games of this type, one of the two roles involves more interaction or is faster-paced and thus more enjoyable than the other role. In such cases, to balance the game and maintain an equal level of player engagement, player roles can switch after each round; the guesser becomes the describer, and the describer becomes the guesser.

Input-agreement games. Representing a generalization of games like Edith Law's TagATune[9] (see Figure 4), they can be described through the following rules:

Initial setup. Two strangers are randomly chosen by the game itself from among all potential players;

Rules. In each round, both players are given inputs that are known by the game (but not by the players) to be the same or different. The players are instructed to produce outputs describing their input, so their partners are able to assess whether their inputs are the same or different. Players see only each other's outputs; and

Winning condition. Both players correctly determine whether they have been given the same or different inputs.

In TagATune, the input is a sound clip, and the output is a series of labels or tags for the clip. The two players achieve the winning condition (and obtain points) only if they both correctly determine whether they have the same input song. Because players want to achieve the winning condition, they each want their partner to be able to determine if their inputs are the same. This means it is in their own best interest to enter accurate outputs that appropriately describe their individual inputs.

To discourage players from randomly guessing whether their inputs are the same, scoring in input-agreement games strongly penalizes incorrect guesses. One way to do this (while maintaining a positive scoring system) is to give an increasing number of points for streaks of correct answers and zero points for incorrect answers.

Increase Player Enjoyment
Perhaps the most important aspect of GWAP is that the output is produced in a way that's designed to be enjoyable. As noted with respect to the ESP Game, players are not directly instructed to enter keywords for a given image. Rather, they are told to type what they think their partner is typing. The fact that people enjoy the game makes them want to continue playing, in turn producing more useful output.

It is important to note that the three basic templates defined earlier describe the basic structure of a GWAP; additional game mechanisms must be added to them to increase player enjoyment. For example, much of the previous work describing game-design principles cites challenge as a key aspect of any successful game.[11,12,14,20] Challenge translates into game features (outlined by Malone[11,12]) like timed response, score keeping, player skill level, high-score lists, and randomness:

Timed response. Setting time limits for game sessions introduces challenge into a game in the form of timed response.[11,12] Players are told to complete a designated number of problem instances within an assigned time limit. If they accomplish it, they may be given extra points for their performance. Timed response is effective for introducing challenge because it establishes an explicit goal that is not trivial for players to achieve if the game is calibrated properly.[11,12] We know from the literature on motivation in psychology and organizational behavior that goals that are both well-specified and challenging lead to higher levels of effort and task performance than goals that are too easy or vague.[10] It is essential that the number of tasks for players to complete within a given time period is calibrated to introduce challenge and that the time limit and time remaining are displayed throughout the game.

Score keeping. One of the most di-

> It is essential that the number of tasks for players to complete within a given time period is calibrated to introduce challenge and that the time limit and time remaining are displayed throughout the game.

rect methods for motivating players is by assigning points for each instance of successful output produced during the game. For the ESP Game,[22] pairs of players are given points for each image for which they successfully agree on a word (which then becomes a label for the image). Using points increases motivation by providing a clear connection among effort in the game, performance (achieving the winning condition), and outcomes (points).[11,12] A score summary following each game also provides players with performance feedback,[10] facilitating progress assessment on score-related goals (such as beating a previous game score and completing all task instances within the set time limit).

Player skill levels. Player skill levels, or "ranks," are another way for game developers to incorporate goal-based motivation into GWAP design. For example, the ESP Game and Peekaboom each have five skill levels players are able to achieve based on the number of points they accumulate. Each newcomer to the game initially has no points and is assigned to the lowest level ("newbie") then has to earn a certain number of points to advance to the next level.

Following each game session, players are shown their current skill level and the number of points needed to reach the next level.[10] Data from the ESP Game indicates that presentation of this skill-level information strongly influences player motivation and behavior. Of the 200,000+ players as of July 2008 with an account on the ESP Game, 42% have scores that fall within 5,000 points of the rank cutoffs. Given that these skill-level point intervals cover less than 2% of the space of possible cumulative scores, the data suggests that many players continue playing just to reach a new rank.

High-score lists. Another method for motivating GWAP play is the use of high-score lists showing the login names and score of the subset of players with the highest number of points over a certain period of time. The score needed by players to be listed on a high-score list varies in terms of difficulty relative to the list's time period, ranging from highest scores achieved in the past game session over the past hour or week all the way to the history

of the game. For example, an hourly high-score list gives players a specific point total to aim for to get onto the list, as well as relatively quick feedback (within the hour) about their progress toward it. A daily high-score list and all-time high-score list define goals of increasing difficulty. These multi-level goals, varying in difficulty, provide strong, positive motivation for extended game play—and related data generation.

Randomness. GWAPs should also incorporate randomness. For example, inputs for a particular game session are typically selected at random from the set of all possible inputs, and players are randomly paired to prevent cheating.

Because inputs are randomly selected, their difficulty varies, thus keeping the game interesting and engaging for expert and novice players alike.[11,12] It also means that every game session involves uncertainty about whether all inputs will be completed within the time limit, adding to the challenge experienced by players.[11,12]

Random partner assignment also ensures the uniqueness of each game session. Anecdotal evidence from the ESP Game[22] suggests that during each game session players develop a sense of their partners' relative skill, a perception that affects their joint performance. The feeling of connection that players can get from these games is one of the factors that motivates repeated play.[18,20]

Output Accuracy
Additional mechanisms must be added to GWAPs beyond the basic template structure to ensure output correctness and counter player collusion. For example, players of the ESP Game might try to circumvent the game's built-in verification mechanism by agreeing prior to the game that for every image they will always type the letter "a"; in this case, they would always match each other, and incorrect data would therefore be entered into the system. We describe generally applicable mechanisms in the following sections that have proved successful in guarding against player collusion and guaranteeing the correctness of the computation across all game templates.

Random matching. GWAPs are

The real measure of utility for a GWAP is therefore a combination of throughput and enjoyability.

meant to be played by hundreds, if not thousands, of people at once, most in distributed locations. Players paired or grouped randomly have no way of knowing their partner's identity so have no easy way to agree ahead of time on any cheating strategy. Thus, under random matching, the probability of two or more cheaters using the same strategy being paired together should be low.

Player testing. Games may randomly present players inputs for which all possible correct outputs are already known. For them, if the output produced by a particular player does not match the known correct outputs, the players should be considered suspicious, and none of their results should be trusted. Depending on the number of "test" inputs presented to players, this strategy can guarantee with high probability that the output is correct. To illustrate, assume half of the inputs given to a player are test inputs. The probability is thus that a new output by the player is correct, given of course that the player is correct on all the test inputs at least 50% of the time, a probability that can be increased through repetition.

Repetition. A game should be designed so it does not consider an output correct until a certain number of players have entered it. This strategy for determining correctness enables any GWAP to guarantee correct output with arbitrarily high probability. As an example, consider an output-agreement game; if for a given input the game accepts an output as correct only after n pairs have entered it, and the game itself knows that each of these n pairs entered a correct output with at least 50% probability (as a result of player testing), then the output is correct with probability of at least $(1–\frac{1}{2}^n)$.

Taboo outputs. For problems in which many different outputs can be associated with one input (such as labeling images with words), ensuring sufficient coverage of the output space is an important consideration. The use of "taboo," or off-limits, outputs provides some guarantee that a larger proportion of all possible outputs will be entered by all players. Taboo outputs are known correct outputs displayed onscreen during game sessions that players are not allowed to enter. They can be taken from correct outputs gen-

erated in previous rounds of the game itself. It is important for the game's designer to randomize which taboo outputs are presented in order to account for potential output-priming effects (in which the particular taboo outputs shown to the players influence the guesses they enter) and ensure wide coverage of all potential outputs for a given input.

Other Design Guidelines

The general schemes we've presented here for designing GWAPs rely on the participation of two players per game session. Now we show that the games can be modified to accommodate single or more than two players.

Prerecorded games. Paired game play makes GWAPs social, meaning that players are able to validate each other's computation. However, two-player games present logistical challenges. For instance, there may be times when an odd number of people want to play a particular game, meaning at least one of them cannot play. In addition, when a game is just beginning to gain popularity, it is difficult for game adminis-

trators to guarantee that many people will be able to play at the same time. We thus recommend that game developers apply a technique—prerecorded game play—introduced by the ESP Game.[22] A dyadic game, normally played by multiples of two players, can be transformed into a single-player game by pairing a single player with a prerecorded set of actions.

In the case of an input-agreement game or output-agreement game (such as the ESP Game), implementing automated players is relatively easy. When two people are playing, the game should simply record every action they make, along with the relative timing of each action. Then, when a single player wishes to play, the system can pair that single player with a prerecorded set of moves.

In inversion-problem games, implementing prerecorded game play is more complex because one of the players (the guesser) must dynamically respond to the other (human) player's actions. Peekaboom, Phetch, and Verbosity have each implemented a single-player version using techniques cus-

tomized to each game.[23–25]

More than two players. The three GWAP templates can be extended to include more than two players; for example, output-agreement games can be extended to incorporate more players by modifying the winning condition such that the first two players who agree on the output are the winners of the round (and granted a higher number of points than the nonwinners). Similarly, the template for inversion-problem games can be extended to incorporate multiple players by substituting an individual guesser with an arbitrary number of players in the role of guesser, all racing to be first to correctly guess the input (winning condition).

These extensions change the nature of the games considerably. Whereas the two-player versions of each template are cooperative in nature (players work together to obtain points), the multiplayer versions are competitive. Cooperative, as well as competitive, games involve advantages and disadvantages. For certain players, competitive games may be more enjoyable than

---

The ESP Game and Verbosity

Two of the most popular GWAPS—ESP and Verbosity—can be played online at www.gwap.com.

The ESP Game has generated millions of labels for random images located throughout the Web. In it, two players are randomly paired for two-and-a-half minutes as they are shown a series of images to label. The game does not directly ask them to label the images. Rather, both players must try to enter the same word as their partner for each image on the screen; neither player can see the partner's words. When both players agree on a word, each is given a new image. The goal is to agree with the partner on words for as many images as possible. The words the players agree on for each image are extremely accurate labels that can be used to improve image search throughout the Web. To increase the quality of these labels, as well as to motivate player engagement, the game

forbids the use of "taboo words" from being entered. In the screenshot (see Figure a), players cannot use the words "dog" or "pillow" when trying to agree on a word with their partner.

Verbosity is a word-guessing game in which two players alternate roles. The describer is given a secret word the guesser

must figure out as quickly as possible. The describer helps the guesser by providing clues about the secret word using sentence templates that must be completed without using the secret word itself. In the example here (see Figure b), the secret word is "sock," and the sentence template "It is a kind of _____"

has been instantiated to the clue "It is a kind of clothing." The describer sees all of the guesser's inputs and indicates which ones are "hot" and which are "cold." The computational purpose of the game is to collect a database of commonsense facts about the secret words (such as "Sock is a kind of clothing").

Figure a: Players of the ESP Game try to guess what their partner is typing on each image.

Figure b: Players of Verbosity enter commonsense facts to help their partner guess a secret word.

their cooperative counterparts. On the other hand, having more players work on the same input is wasteful in terms of "computational efficiency," an important criterion for evaluating the utility of a given game.

GWAP Evaluation

How might a game's performance be judged successful? Given that two different GWAPs solve the same problem, which is best? We describe a set of metrics for determining GWAP success, including throughput, lifetime play, and expected contribution.

Game efficiency and expected contribution. If we treat games as if they were algorithms, efficiency would be a natural metric of evaluation. There are many possible algorithms for any given problem, some more efficient than others. Similarly, many possible GWAPs are available for any given problem. In order to choose the best solution to a problem we need a way to compare the alternatives in terms of efficiency. Efficiency of standard algorithms is measured by counting atomic steps. For instance, QuickSort is said to run in *O(n log n)* time, meaning it sorts a list of *n* elements in roughly *n log n* computational steps. In the case of GWAPs, the notion of what constitutes a computational step is less clear. Therefore, we must be able to define efficiency through other means.

First, we define the throughput of a GWAP as the average number of problem instances solved, or input-output mappings performed, per human-hour. For example, the throughput of the ESP Game is roughly 233 labels per human-hour.[22] This is calculated by examining how many individual inputs, or images, are matched with outputs, or labels, over a certain period of time.

Learning curves and variations in player skill must be considered in calculating throughput. Most games involve a certain type of learning, meaning that with repeated game sessions over time, players become more skilled at the game. For the game templates we described earlier, such learning can result in faster game play over time. To account for variance in player skill and changes in player speed over time as a result of learning, we define throughput as the average number of problem instances solved per human-hour. This

average is taken over all game sessions through a reasonably lengthy period of time and over all players of the game.

Games with higher throughput should be preferred over those with lower throughput. But throughput is not the end of the story. Because a GWAP is a game, "fun" must also be included. It does not matter how many problem instances are addressed by a given game if nobody wants to play. The real measure of utility for a GWAP is therefore a combination of throughput and enjoyability.

Enjoyability is difficult to quantify and depends on the precise implementation and design of each game. Even seemingly trivial modifications to a game's user interface or scoring system can significantly affect how enjoyable it is to play. Our approach to quantifying this elusive measure is to calculate and use as a proxy the "average lifetime play" (ALP) for a game. ALP is the overall amount of time the game is played by each player averaged across all people who have played it. For instance, on average, each player of the ESP Game plays for a total of 91 minutes.

"Expected contribution" is our summary measure of GWAP quality. Once a game developer knows on average how many problems are solved per human-hour spent in the game (throughput) and how much time each player can be expected to spend in a game (ALP), these metrics can be combined to assess each player's expected contribution. Expected contribution indicates the average number of problem instances a single human player can be expected to solve by playing a particular game. Developers can then use this measure as a general way of evaluating GWAPs. We define the three GWAP metrics this way:

Throughput = average number of problem instances solved per human-hour;

ALP = average (across all people who play the game) overall amount of time the game will be played by an individual player; and

Expected contribution = throughput multiplied by ALP.

Although this approach does not capture certain aspects of games (such as "popularity" and contagion, or word of mouth), it is a fairly stable measure of a game's usefulness. Previous work

in the usability tradition on measuring fun and game enjoyment has suggested the usefulness of self-report questionnaire measures.[7,14] However, a behavioral measure (such as throughput) provides a more accurate direct assessment of how much people play the game and, in turn, how useful the game is for computational purposes.

Finally, a GWAP's developers must verify that the game's design is indeed correct; that is, that the output of the game maps properly to the particular inputs that were fed into it. One way to do this (as with the ESP Game, Peekaboom, Phetch, and Verbosity) is to analyze the output with the help of human volunteers. We have employed two techniques for this kind of output verification: comparing the output produced in the game to outputs generated by paid participants (rather than game players)[22] and having independent "raters" evaluate the quality of the output produced in the game.[22] Output from a GWAP should be of comparable quality to output produced by paid subjects.

Conclusion

The set of guidelines we have articulated for building GWAPs represents the first general method for seamlessly integrating computation and gameplay, though much work remains to be done. Indeed, we hope researchers will improve on the methods and metrics we've described here.

Other GWAP templates likely exist beyond the three we have presented, and we hope future work will identify them. We also hope to better understand problem-template fit, that is, whether certain templates are better suited for some types of computational problems than others.

The game templates we have developed thus far have focused on similarity as a way to ensure output correctness; players are rewarded for thinking like other players. This approach may not be optimal for certain types of problems; in particular, for tasks that require creativity, diverse viewpoints and perspectives are optimal for generating the broadest set of outputs.[17] Developing new templates for such tasks could be an interesting area to explore.

We would also like to understand what kinds of problems, if any, fall outside the GWAP approach. The games

we have designed so far have focused on problems that are easily divided into subtasks. The "bite-size" nature of these games adds to their popularity and appeal to casual gamers in particular, since such players typically go for games they can play "just one more time" without having to make too much of a time commitment.

The GWAP approach represents a promising opportunity for everyone to contribute to the progress of AI. By leveraging the human time spent playing games online, GWAP game developers are able to capture large sets of training data that express uniquely human perceptual capabilities. This data can contribute to the goal of developing computer programs and automated systems with advanced perceptual or intelligence skills.

Acknowledgment

References
1. Carroll, J.M. and Thomas, J.M. Fun. *ACM SIGCHI Bulletin 19*, 3 (Jan. 1988), 21–24.
2. Chao, D. Computer games as interfaces. *Interactions 11*, 5 (Sept.–Oct. 2004), 71–72.
3. Chao, D. Doom as an interface for process management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seattle, Mar. 31–Apr. 5). ACM Press, New York, 2001, 152–157.
4. Fails, J.A. and Olsen, D.R. Interactive machine learning. In *Proceedings of the Eighth International Conference on Intelligent User Interfaces* (Miami, Jan. 12–15). ACM Press, New York, 2003, 39–45.
5. Federoff, M. *Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games.* Unpublished thesis, Indiana University, Bloomington; www.melissafederoff.com/thesis.html.
6. Giles, J. Internet encyclopaedias go head to head. *Nature 438* (Dec. 15, 2005), 900–901.
7. Hassenzahl, M., Beu, A., and Burmeister, M. Engineering joy. *IEEE Software 18*, 1 (Jan.–Feb. 2001), 70–76.
8. Laurel, B.K. Interface as mimesis. In *User-Centered System Design: New Perspectives on Human-Computer Interaction*, D.A. Norman and S.W. Draper, Eds. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, 1986, 67–85.
9. Law, E.L.M., von Ahn, L., Dannenberg, R.B., and Crawford, M. TagATune: A game for music and sound annotation. In *Proceedings of the Eighth International Conference on Music Information Retrieval* (Vienna, Austria, Sept. 23–30). Austrian Computer Society, Vienna, Austria, 2007, 361–364.
10. Locke, E.A. and Latham, G.P. *A Theory of Goal Setting and Task Performance.* Prentice Hall, Englewood Cliffs, NJ, 1990.
11. Malone, T.M. Heuristics for designing enjoyable user interfaces: Lessons from computer games. In *Proceedings of the Conference on Human Factors in Computing Systems* (Gaithersburg, MD, Mar. 15–17). ACM Press, New York, 1982, 63–68.
12. Malone, T.M. What makes things fun to learn? Heuristics for designing instructional computer games. In *Proceedings of the Third ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems* (Palo Alto, CA, Sept. 18–19). ACM Press, New York, 1980, 162–169.
13. Mokka, S., Väätänen, A., Heinilä, J., and Välkkynen, P. Fitness computer game with a bodily user interface. In *Proceedings of the Second International Conference on Entertainment Computing* (Pittsburgh, PA, May 8–10). Carnegie Mellon University, Pittsburgh, PA, 2003, 1–3.
14. Pagulayan, R., Keeker, K., Wixon, D., Romero, R., and Fuller, T. User-centered design in games. *In The Human-Computer Interaction Handbook: Fundamentals, Evolving Techniques and Emerging Applications*, J.A. Jacko and A. Sears, Eds. Lawrence Erlbaum Associates, Mahwah, NJ, 2003, 883–905.
15. Richards, C. Teach the world to twitch: An interview with Marc Prensky, CEO and founder Games2train.com. *Futurelab* (Dec. 2003); www.futurelab.org.uk/resources/publications_reports_articles/web_articles/Web_Article578.
16. Shneiderman, B. Designing for fun: How can we design user interfaces to be more fun? *Interactions 11*, 5 (Sept.–Oct. 2004), 48–50.
17. Steiner, I. *Group Process and Productivity.* Academic Press, New York, 1972.
18. Stork, D.G. and Lam C.P. Open mind animals: Ensuring the quality of data openly contributed over the World Wide Web. In *Learning from Imbalanced Data Sets: Papers from the AAAI Workshop (Technical Report WS-00-05)*. (Austin, TX, July 30–Aug. 1). American Association for Artificial Intelligence, Menlo Park, CA, 2000, 4–9.
19. Stork, D.G. The Open Mind Initiative. *IEEE Intelligent Systems & Their Applications 14*, 3 (May–June 1999), 19–20.
20. Sweetser, P. and Wyeth, P. GameFlow: A model for evaluating player enjoyment in games. *ACM Computers in Entertainment 3*, 3 (July 2005), 3.
21. Tsang, M., Fitzmaurice, G., Kurtenbach, G., and Khan, A. Game-like navigation and responsiveness in non-game applications. *Commun. ACM 46*, 7 (July 2003), 57–61.
22. von Ahn, L. and Dabbish, L. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vienna, Austria, Apr. 24–2). ACM Press, New York, 2004, 319–326.
23. von Ahn, L., Ginosar, S., Kedia, M., and Blum, M. Improving image search with Phetch. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (Honolulu, Apr. 15–20). IEEE Press, New York, 2007, IV–1209–IV–1212.
24. von Ahn, L., Kedia, M., and Blum, M. Verbosity: A game for collecting common-sense knowledge. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montreal, Apr. 22–27). ACM Press, 2007, 75–78.
25. von Ahn, L., Liu, R., and Blum, M. Peekaboom: A Game for locating objects in images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montreal, Apr. 22–27). ACM Press, New York, 2006, 55–64.
26. Webster, J. Making computer tasks at work more playful: Implications for systems analysts and designers. In *Proceedings of the SIGCPR Conference on Management of Information Systems Personnel* (College Park, MD, Apr. 7–8). ACM Press, New York, 1988, 78–87.

Luis von Ahn (lav@andrew.cmu.edu) is an assistant professor in the Computer Science Department at Carnegie Mellon University, Pittsburgh, PA.

Laura Dabbish (dabbish@andrew.cmu.edu) is an assistant professor of information technology and organizations in the Heinz School, with a joint appointment in the Human-Computer Interaction Institute in the School of Computer Science at Carnegie Mellon University, Pittsburgh, PA.