

Exploiting Puzzle Diversity in Puzzle Selection for ESP-like GWAP Systems *

Yu-Song Syu[†], Hsiao-Hsuan Yu[†], and Ling-Jyh Chen^{†‡}

[†]Research Center for Information Technology Innovation, Academia Sinica

[‡]Institute of Information Science, Academia Sinica

{yssyu, hhsuanyu, ccljjj}@iis.sinica.edu.tw

Abstract

The ESP game belongs to the genre called *Games with a Purpose (GWAP)*, which leverage people’s desire to be entertained and also outsource certain steps of the computational process to humans. The productivity of ESP-like GWAP systems depends to a great extent on the puzzle selection strategy used in the system. Although traditional approaches seek to determine the optimal number of agreements reached in each puzzle, they may be affected by the equality of outcomes issue because they ignore the differences among puzzles. In this paper, using realistic game traces, we define the puzzle diversity issue and propose a novel approach, called the *Adaptive Puzzle Selection Algorithm (APSA)*, to promote equality of opportunity in ESP-like GWAP systems. We also introduce a data structure called the *Weight Sum Tree (WST)* to reduce the computational complexity of the proposed scheme and facilitate its implementation in real-world systems. Using a comprehensive set of simulations, we evaluate the APSA scheme against the traditional *OPSA* scheme, and demonstrate that APSA can better accommodate the differences among puzzles in ESP-like GWAP systems.

1 Introduction

Games With A Purpose (GWAP) [30, 32] represent a new paradigm of applications that leverage people’s desire to be entertained and also outsource certain steps of the computational process to humans [19, 21, 29]. By exploiting “human cycles” in computation, GWAP motivate people to play voluntarily, and also produce useful metadata as a by-product. The genre has shown promise in solving a variety of problems, such as image annotation [26, 31, 36], audio annotation [5, 7, 8, 22], and commonsense reasoning

[23, 35], which computer programs have been unable to resolve completely thus far.

Various GWAP systems have been proposed in recent years [23, 26, 31, 33, 35, 36]. Among them, the ESP Game [31] was the first to successfully realize the advantages of GWAP systems. The rationale behind the ESP game is to motivate people to label images because it is fun. It has been shown that the image labels collected through the ESP game are typically of good quality. Moreover, the game results allow more accurate image retrieval, help users block inappropriate (e.g., pornographic) images, and improve web accessibility (e.g., the labels can help visually impaired people surf web pages [11]).

To be effective, the ESP game tries to collect outcomes with the largest possible aggregated score for each puzzle (image). It also needs as many distinct puzzles as possible to be played. There is a trade-off between these two goals. On the one hand, the system tries to collect as many labels as possible for each puzzle, and this results in the playing of fewer distinct puzzles; on the other hand, the system prefers that each puzzle is played only once, so that the maximum number of puzzles can be played. The problem has been formulated as a scheduling problem, and the *Optimal Puzzle Selection Algorithm (OPSA)* has been proposed to determine the optimal number of agreements required for all puzzles based on an analytical model [14, 15]. However, the OPSA scheme does not consider the differences among puzzles (some puzzles are more productive, and some are hard to solve), which may result in the *equality of outcomes* problem. This finding motivates us to devise appropriate mechanisms that can better accommodate *puzzle diversity*, i.e., the differences among puzzles, in ESP-like GWAP systems.

The contribution of this work is two-fold. First, using realistic game traces, we identify the *puzzle diversity* issue that is common in ESP-like GWAP systems, and propose a novel approach, called the *Adaptive Puzzle Selection Algorithm (APSA)*, to cope with puzzle diversity by promoting *equality of opportunity*. The scheme selects the puzzle to be played based on its previous play history. Second, we pro-

*This research was supported in part by the National Science Council of Taiwan under Grants: NSC 98-2221-E-001-014-MY3 and NSC 98-2631-H-001-013.

pose a data structure, called the *Weight Sum Tree* (WST), to reduce the computational complexity and facilitate the implementation of the APSA scheme in real-world systems. Using a comprehensive set of simulations, we evaluate the proposed scheme against the OPSA scheme and show that it is more effective in terms of the number of agreements reached and the system gain. Finally, the APSA scheme is simple and shows promise for use in the design and implementation of future ESP-like GWAP systems.

The remainder of this paper is organized as follows. Section 2 contains a review of related works on GWAP systems and the ESP game. In Section 3, we present the proposed APSA approach, and discuss a number of implementation issues. In Section 4, we detail and analyze the experiment results. We then summarize our conclusions in Section 5.

2 Background

2.1 GWAP Overview

The concept of GWAP was pioneered by Luis von Ahn and his colleagues [30]. In recent years, a substantial and increasing amount of research effort has been invested in the area, and several GWAP systems have been developed for a variety of purposes [3, 5, 7–10, 13, 17, 22–24, 26, 28, 31, 33–36, 38]. The online ESP Game¹ [31] was the first GWAP system, and it was subsequently adopted as the *Google Image Labeler* [3]. The game is fast-paced, enjoyable, and competitive. As of July 2008, more than 200,000 players had contributed more than 50 million labels. Each player plays for a total of 91 minutes on average, and the throughput is about 233 labels per player per hour (i.e., one label every 15 seconds) [32]. Moreover, it has been shown that the collected labels facilitate more accurate image retrieval, help users block inappropriate images, and improve web accessibility.

In addition, the *Peekaboom* system [36] helps users determine the location of objects in images; while the *Squigl* system [4] and the *LabelMe* system [26] provide complete outlines of the objects in an image. *Phetch* [33, 34] produces image descriptions that improve web accessibility and image searches, while the *Matchin* system [4] helps image search engines rank images based on which ones look the best. The concept of the ESP Game has been applied to other problems. For instance, the *Herd It* [5, 8], *Major Miner* [7], and *TagATune* [22] systems, which provide annotation for sounds and music, can improve audio searches. The *Verbosity* system [35] and the *Common Consensus* system [23] collect “common-sense” knowledge

¹The game is called ‘ESP’ because the players have to work together to solve the tasks without talking to each other, i.e., by using *Extra-Sensory Perception* (ESP) [31].

that is valuable for commonsense reasoning and enhancing the design of interactive user interfaces. In [10], Bennett et al. propose a system that collects individual user preferences for image-search results, and then extracts consensus rankings from the preferences for the results of a query. The *Context-Aware Recognition Survey* (CARS) system [38] uses ubiquitous sensors to monitor activities in the home, while [9, 13, 17, 24] employ mobile social gaming for geospatial tagging. Moreover, [28] applies human computation to ontology alignment and web content annotation for the Semantic Web using a set of games, such as *OntoPronto*, *SpotTheLinks*, *OntoTube*, and *OntoBay*. Finally, Shenoy and Tan [27] showed that it is possible to design environments in which humans cannot avoid processing some of the tasks (and producing some useful outcomes), even though they are not actively trying to do so.

In addition to designing new GWAP systems, several studies have investigated the performance aspect of human computation [16, 18, 20, 32, 37]. For example, Ho et al. [18] proposed solving the *coalition* problem by integrating both collaborative and competitive elements in image labeling games. Gentry et al. [16] proposed a framework of vote-based human computation and provided a probabilistic analysis of the reliability of the voting mechanism and design principles on the payout function. In [37], Weber et al. presented a machine learning-based model that can play the ESP game without looking at the image. Based on the model, the authors proposed an enhanced scoring system for the ESP game to encourage users to contribute less predictable labels and thereby improve the quality of the collected labels. Jain and Parkes [20] applied game theoretic analysis to the ESP game. They investigated the equilibrium behavior under different incentive mechanisms and provided guidelines for the design of such mechanisms. Von Ahn [32] proposed a set of evaluation metrics, namely, throughput, lifetime play, and expected contribution, to determine whether ESP-like GWAP systems are successful.

2.2 ESP-like GWAP Systems

In the ESP game, when a user logs into the system, he/she is automatically matched with a random partner. The two players do not know each other’s identity as they cannot communicate. Initially, a randomly selected image (called a puzzle) is displayed to both players simultaneously. The players then input possible words to label the image until an *agreement* is reached (i.e., the same word is entered by both players); or they can decide to pass over a puzzle if they think it is too difficult. Once an agreement has been reached, a bonus score is awarded to each player based on the *quality* of the agreed word. In practice, the quality of a word is measured by its popularity; generally, words that are more popular receive lower scores. After the players

agree on a word, they are shown another puzzle. In each ESP game, they have two and a half minutes to label 15 images.

In this work, we define *ESP-like GWAP Games* as games that satisfy the following four criteria: 1) each game involves at least two players; 2) each game lasts for at least one round; 3) the system selects a puzzle to play in each round; and 4) the result of a round is either a success (i.e., an agreement is reached) or a failure (i.e., the players decide to pass the game, or timeout is reached). In addition to the ESP game, several GWAP systems belong to this category, e.g., *Google Image Labeler* [3], *Peekaboom* [36], *Verbosity* [35], *TagATune* [22], and *Herd It* [5, 8].

To be effective, the ESP game should satisfy two goals. First, it should collect as many labels as possible for each puzzle, so that fewer distinct puzzles will be played. Second, each puzzle should only be played once in order to maximize the number of puzzles played. In [14, 15], Chen et al. proposed a *system gain* metric for assessing the performance of GWAP systems, and designed the *Optimal Puzzle Selection Algorithm* (OPSA) based on an analytical model. The metric is comprised of two parts: *agreement throughput* and *agreement quality*, and the OPSA scheme seeks to determine the optimal number of the agreements required for all puzzles in order to maximize the system gain. The main drawback of the OPSA scheme is that it does not consider the differences among puzzles. Consequently, it suffers from the *equality of outcomes* issue caused by puzzle diversity.

3 The Proposed Approach

We present a novel approach called the *Adaptive Puzzle Selection Algorithm* (APSA) for ESP-like GWAP systems. The approach was inspired by the *Additive Increase Multiplicative Decrease* (AIMD) model of the *Transmission Control Protocol* (TCP) [25], the most widely used transport protocol on the Internet. The rationale behind APSA is that it promotes *equality of opportunity* by considering puzzle diversity, instead of developing *equality of outcomes* like traditional approaches. Recall that the latter ignore the differences among puzzles. In the following subsections, we present the proposed APSA scheme and discuss related implementation issues.

3.1 The APSA Approach

We assume there are K puzzles in the system, and that the system selects a puzzle based on the weighted values of the puzzles in each game round. Let w_k denote the weighted value of the k -th puzzle, and let p_k denote the probability that the k -th puzzle will be selected in the next game round.

We obtain the value of p_k by²

$$p_k = \frac{w_k}{\sum_{i=1}^K w_i}. \quad (1)$$

When a new game round is initiated, the system selects the puzzle to be played based on the values of p_k for $1 \leq k \leq K$. The steps of the decision process are as follows. First, the system determines a random number r between 0 and 1 (inclusive) using a uniform random number generator. Then, the i -th puzzle is selected if it satisfies the following criteria:

$$\frac{\sum_{k=1}^{k=i-1} w_k}{\sum_{k=1}^{k=K} w_k} < r \leq \frac{\sum_{k=1}^{k=i} w_k}{\sum_{k=1}^{k=K} w_k}. \quad (2)$$

The weighted value of each puzzle is set to 1 initially; then the value is updated using the AIMD model. There are two possible scenarios.

- If *agreements are reached* in a game round, we assume there is a very high likelihood that the puzzle will also yield agreements in the next game round (i.e., the Pareto principle [12]). Hence, we increase its weighted value w_k by one (i.e., an additive increase).
- If *no agreements are reached* in a game round, we assume the puzzle is too difficult for the players, and we reduce the value of w_k by half (i.e., a multiplicative reduction).

More precisely, the value of w_k is determined by

$$w_k = \begin{cases} 1 & \text{the initial value,} \\ w_k + 1 & \text{if agreements are reached,} \\ \frac{w_k}{2} & \text{if no agreements are reached.} \end{cases} \quad (3)$$

Initially, all puzzles have the same likelihood of being selected in a game round (since the initial weight is set to 1 for all puzzles, i.e., $p_k = \frac{1}{K}$ for $k = 1 \dots K$). As the game proceeds, p_k is updated based on the value of w_k . That is, the more productive a puzzle is (in terms of the number of game rounds that yield agreements), the higher the probability it will be selected in the next game round. As a result, compared to the OPSA scheme, the APSA scheme is better able to deal with *puzzle diversity* (i.e., the differences among puzzles), and each puzzle will be played in a different number of rounds depending on its productivity.

3.2 Implementation Method

As mentioned earlier, the APSA scheme uses the AIMD model to adapt each puzzle's probability of being selected to

²In the OPSA scheme, the value of w_k is set to a large constant if the number of agreements reached in the k -th puzzle is larger than 0 and smaller than the optimal number; otherwise, it is set to 0.

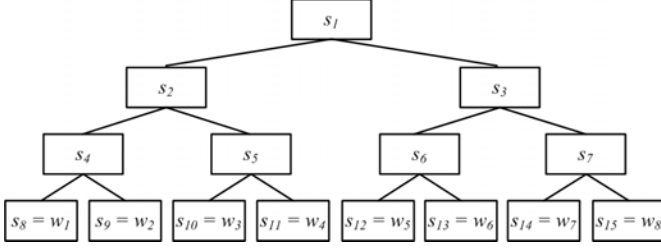


Figure 1. An example of a Weight Sum Tree ($K = 8$)

play in a game round. However, the proposed scheme has two drawbacks: 1) it needs to store the information about each puzzle (i.e., w_k), and the storage complexity is approximately $O(K)$; and 2) it has to update the values of w_k and p_k frequently, and the computational complexity is approximately $O(K)$. As the complexity increases linearly with the number of the puzzles, it may be too costly for real-world systems. To resolve the scalability issue, we build a *complete binary tree of partially weighted sums*, called a *Weight Sum Tree (WST)*, to implement the APSA scheme in ESP-like GWAP systems.

For example, as shown in Figure 1, if there are eight puzzles in the system (i.e., $K = 8$), and the k -th puzzle's weight is w_k , we build a WST to store the *partially weighted sums* of the puzzles. In this case, the height of the tree (h) is 4 (i.e., $\log_2 K + 1 = 4$). Note that if the value of K is not a *positive integer power of 2*, it is necessary to create $2^h - K$ *virtual puzzles* whose weights are all equal to zero in order to ensure the completeness of the WST. The nodes are ordered from the top to the bottom and from left to right. Let s_i denote the i -th node in the tree; then we can obtain the value of s_i by

$$s_i = \begin{cases} w_{i-2^{h-1}+1} & , \text{ when } 2^{h-1} \leq i < 2^h; \\ s_{2i} + s_{2i+1} & , \text{ when } 0 < i < 2^{h-1}. \end{cases} \quad (4)$$

There are three cases in which we have to update the weight sum tree.

1. *After a puzzle is played in a game round* (say, the k -th puzzle), we update its weight using Equation 4. The *update volume* Δw_k is obtained by subtracting the original weight from the new weight. Then, we update each ancestor node of the puzzle from the bottom to the top of the WST by adding Δw_k . Since the number of nodes that must be updated is equal to the height of the tree, the computational complexity of this case is $O(h)$ (i.e., $O(\log K)$).
2. *After a puzzle has been removed from the system* (say, the k -th puzzle), we set $\Delta w_k = -w_k$ and change the

Algorithm 1 The proposed puzzle selection implementation based on the APSA scheme and the weight sum tree data structure.

- 1: **Function** *Puzzle_Selection*(k, r)
 - 2: **if** $k \geq 2^{h-1}$ **then**
 - 3: Return the $(k - 2^{h-1} + 1)$ th puzzle;
 - 4: **end if**
 - 5: **if** $r \leq \frac{s_{2k}}{s_1}$ **then**
 - 6: *Puzzle_Selection*($2k, r$);
 - 7: **else**
 - 8: *Puzzle_Selection*($2k + 1, r - \frac{s_{2k}}{s_1}$);
 - 9: **end if**
-

value of w_k to 0 (i.e., the k -th puzzle now becomes a virtual puzzle). Then, we update each ancestor node of the puzzle from the bottom to the top of the WST by adding Δw_k . The computational complexity of this case is also $O(\log K)$.

3. *After adding a new puzzle to the system*, we set the weight of the puzzle to 1 (i.e., the default value). There are two cases where the weight sum tree must be updated: 1) if there are *virtual puzzles* in the tree, we replace the first virtual puzzle (i.e., the leftmost leaf whose weight is equal to zero) with the new puzzle, and update its ancestor nodes from the bottom to the top of the tree accordingly; or 2) we create $K - 1$ *virtual puzzles* and rebuild the weight sum tree (i.e., the total number of the puzzles now becomes $2K$). The computational complexity of these two cases is $O(\log K)$ and $O(K)$ respectively.

Under the WST data structure, a puzzle is selected to play in each game round. The steps of the selection process are as follows. First, the system uses a uniform random number generator to determine a random number between 0 and 1 (inclusive). Then, it calls the *Puzzle_Selection*(k, r) function, with the parameters k equal to one and r equal to the newly generated random number, as shown in Algorithm 1. The *Puzzle_Selection*() function traverses the weight sum tree from the root to the leaf node and selects the leaf node (say, the i -th puzzle) such that Equation 2 is satisfied. Note that, under the WST data structure, the computational complexity is reduced from $O(K)$ to $O(\log K)$, but the storage complexity remains $O(K)$.

4 Evaluation

We evaluated the APSA scheme via trace-based simulations. The game trace was collected by the *ESP Lite* system³ [2], which is publicly available to the research com-

³Note that, although the game data set of the ESP game has been released [1], it only contains the agreement words of each puzzle. It does

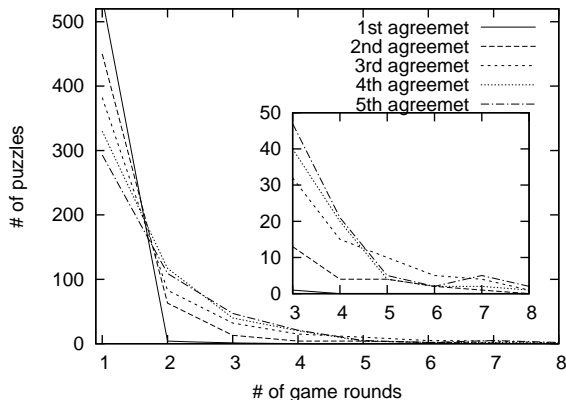


Figure 2. Comparison of the number of game rounds required to reach an agreement.

munity [6]. *ESP Lite* is a ‘quasi’ ESP Game that embeds various puzzle selection algorithms (namely, RPSA, FPSA, and OPSA [15]). It records the complete game trace, including the time required by each game round and the result (either an agreement or a pass). The trace was recorded over a one-month period (from 2009/03/09 to 2009/04/09), and the OPSA scheme was used in 1,444 games comprised of 6,326 game rounds. In total, 575 distinct puzzles were played and 3,418 agreements were reached. We used C programming language to implement the APSA scheme, and compared the system performance, in terms of the puzzle pass rate, agreement throughput, and system gain, with that of the OPSA scheme. In the following sub-sections, we present an analysis and discussion of the experiment results.

4.1 Analysis of Puzzle Diversity

Using the *ESP Lite* game trace, we investigate puzzle diversity, i.e., the difficulty of reaching an agreement in different puzzles. Figure 2 compares the distribution of the number of game rounds required to reach the i -th agreement in the dataset. Note that, since players may pass a puzzle if they feel it is too difficult, a puzzle may have to be played in several rounds to reach an agreement. Intuitively, the greater the difficulty of a puzzle, the larger the number of game rounds required to reach an agreement.

The results in Figure 2 show that the number of puzzles for which an agreement is reached in the first game round decreases with the sequence order of the agreements. Specifically, in one game round, 536 puzzles can reach the first agreement, 450 can reach the second agreement, 382

not provide the detailed play history (i.e., the number of play rounds of each puzzle, the pass rate of each puzzle, and play time of each round). As a result, the dataset is insufficient for this study, so we use the *ESP Lite* dataset instead.

can reach the third agreement, 330 can reach the fourth agreement, and 293 can reach the fifth agreement. The results confirm our intuition that *it is more difficult to reach the $(i + 1)$ -th agreement than the i -th agreement in a puzzle.*

Moreover, we observe that, if more than one game round is required to yield an agreement, the number of puzzles increases with the sequence order of the agreements. For instance, if a puzzle requires exactly three game rounds to reach an agreement, one puzzle can reach the first agreement, 13 can reach the second agreement, 32 can reach the third agreement, 40 can reach the fourth agreement, and 47 can reach the fifth agreement. The results demonstrate that *differences exist among the puzzles.* While most puzzles can produce an agreement easily in the first round, it becomes increasingly difficult for puzzles to yield agreements in the second, third, fourth and fifth rounds. Hence, it is important to consider *puzzle diversity* in the design of puzzle selection algorithms. The ideal solution involves promoting *equality of opportunity*, instead of *equality of outcomes*, in order to accommodate the differences among puzzles.

4.2 Simulation Results

In the first set of simulations, we compare the number of game rounds that were passed under the APSA and OPSA schemes. The cumulative distribution function (CDF) curves of the simulation results are shown in Figure 3. Under the OPSA scheme, 83% of the puzzles experience two passes or less; and the percentage increases to 92% under the APSA scheme. Note that, the larger the number of game rounds passed, the smaller the number of agreements collected. Hence, the results demonstrate that the APSA scheme is superior to the OPSA scheme in terms of reducing the number of game rounds that are passed.

Figure 4 compares the number of distinct puzzles played (N_p) and the number of puzzles for which there was at least one agreement (N_t) under the two schemes with various numbers of game rounds. The results demonstrate that the APSA scheme yields larger values of N_p and N_t consistently. Moreover, we observe that, after 2,500 game rounds, the value of N_t in the APSA scheme is comparable to the value of N_p in the OPSA scheme. The reason is that, under the APSA scheme, the system tends to avoid selecting *difficult* puzzles (i.e., puzzles that have a history of being passed) by reducing their weights. In other words, APSA favors puzzles that have not been played before or puzzles that are more likely to yield new agreements. As a result, it is more effective than the OPSA scheme in reducing the number of game rounds that are passed, i.e., it increases the number of the puzzles that reach agreements.

The results in Figure 4 also show that, unlike the APSA scheme, the curves of N_T and N_P contain plateaus when the OPSA scheme is used. This is because, for all puz-

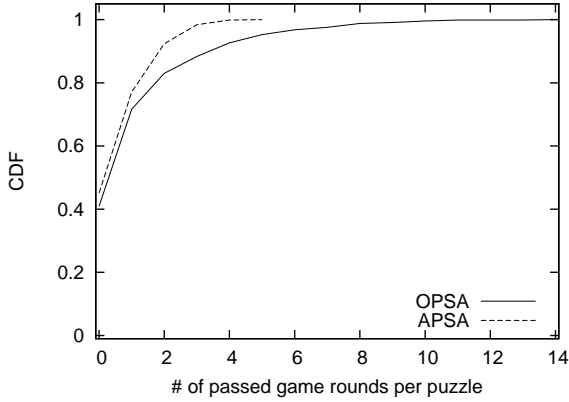


Figure 3. The CDF curves of the average number of the rounds passed per puzzle under the APSA and OPSA schemes.

zles, the OPSA scheme seeks to apply a *global optimal value* as the number of agreements that must be reached in a puzzle. As a result, when the optimal value is updated (i.e., increased by one), the OPSA scheme assigns *old puzzles* first (i.e., puzzles that have been played before), so that they can reach the new number of required agreements. It then assigns *fresh puzzles* (i.e., puzzles that have not been played before) repeatedly until the required number of agreements is achieved. The first case causes the plateaus in the OPSA curves (i.e., the periods when neither N_T nor N_P increases); and the second case results in a nearly linear increase in the curves. Moreover, the length of the plateau period grows with each occurrence, because the number of old puzzles increases in tandem with the occurrences.

Next, we evaluate the efficiency of the APSA and OPSA schemes in terms of per-puzzle throughput, i.e., the number of agreements reached over the total time required for each puzzle. From the results shown in Figure 5, we observe that 90% of the puzzles achieve a throughput below 0.05 agreements/second under the OPSA scheme, but the percentage drops to 80% under the APSA scheme. The results demonstrate that the APSA scheme is more efficient than the OPSA scheme, because it yields more agreements with better per-puzzle throughput.

Figure 6 shows the distribution of the number of agreements reached in the system under the two puzzle selection schemes. The results demonstrate that, under the OPSA scheme, 17% of the puzzles do not yield any agreements, and more than 75% of the puzzles reach the same number of agreements (i.e., 5 agreements). In contrast, under the APSA scheme, 19% of the puzzles do not reach any agreements, and the number of the agreements reached in the remaining puzzles is almost uniformly distributed. This

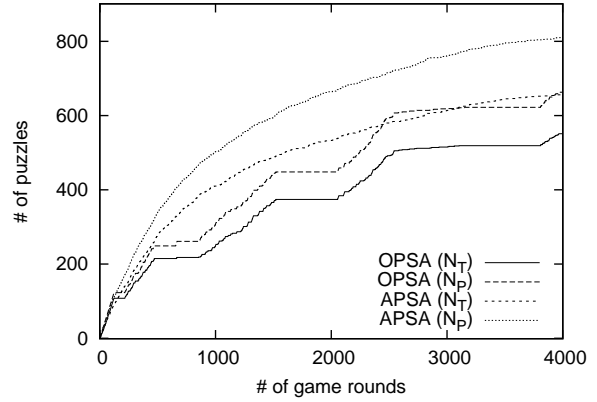


Figure 4. Comparison of the number of puzzles played (N_P) and the number of puzzles with agreements (N_T) under the APSA and OPSA schemes.

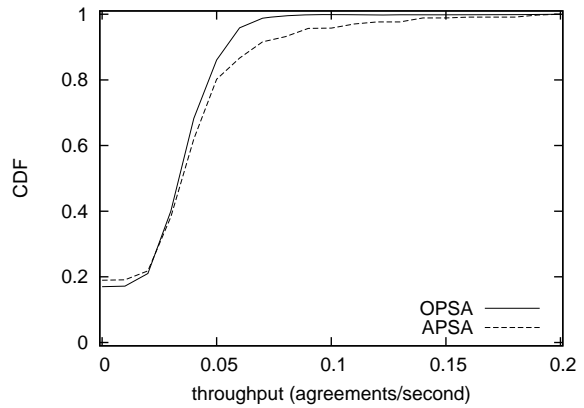


Figure 5. The CDF curves of the agreement throughput per puzzle under the APSA and OPSA schemes.

is because the OPSA scheme emphasizes *equality of outcomes*, and it requires that all solved puzzles have the same number of agreements. The APSA scheme, on the other hand, emphasizes *equality of opportunity*, and it adjusts the probability that a puzzle will be assigned in the next round based on its previous play history (i.e., the probability is higher if the puzzle was more productive in the past). Thus, the results demonstrate that the APSA scheme can better accommodate puzzle diversity, i.e., the differences among puzzles, than the OPSA scheme.

Finally, using the *system gain* metric [14, 15], we compare the system gain achieved by the APSA and the OPSA schemes in the simulations. From the results shown in Figure 7, we observe that the APSA scheme always achieves a

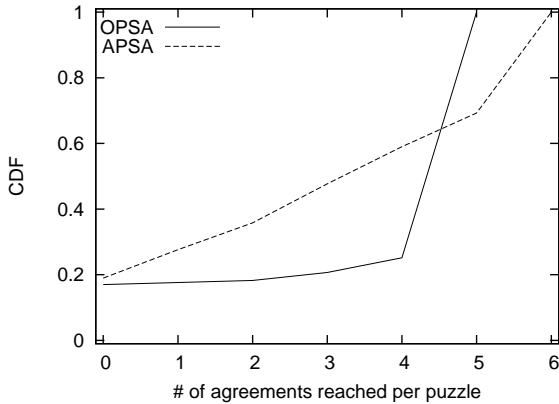


Figure 6. The CDF distribution of the number of agreements reached per puzzle under the APSA and OPSA schemes.

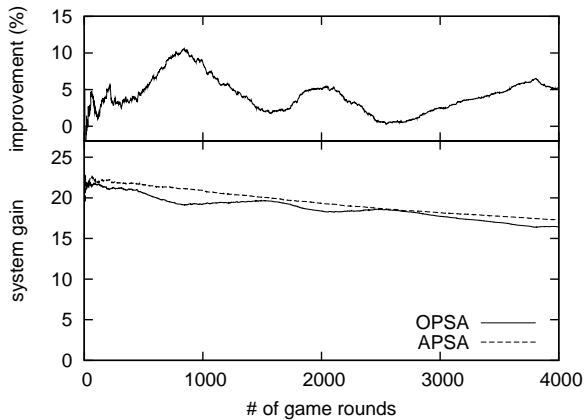


Figure 7. Comparison of the system gain under the APSA and OPSA schemes.

better system gain than the OPSA scheme (about 5% improvement). Note that, the system metric considers two aspects of the system’s performance: *the average time required for each puzzle (including the passed rounds) and the average score of the agreements reached in each puzzle*. Since the APSA scheme is effective in reducing the number of rounds passed in the system, it yields a higher value in the first part of the metric, and thus improves the system gain. The system gain could be improved further by modifying the second part of the metric (e.g., by introducing *competition* into the system [18]). However, we defer consideration of this issue to a future work.

5 Conclusion

In this paper, we investigate the puzzle selection problem in ESP-like GWAP systems. Using realistic game traces, we find that *puzzle diversity* is common in such systems; hence, we argue that it is important to consider puzzle diversity when designing puzzle selection algorithms. To this end, we propose an approach called the *Adaptive Puzzle Selection Algorithm* (APSA), which is based on the *Additive Increase Multiplicative Decrease* (AIMD) model. In addition, we introduce a data structure called the *Weight Sum Tree* (WST) to reduce the computational complexity of APSA in real-world applications. Using a comprehensive set of simulations, we evaluate the proposed scheme against the OPSA scheme, and show that it achieves a better system gain and it is more effective in reducing the number of game rounds passed by players. The results demonstrate that the APSA scheme can better accommodate the *individual differences* among puzzles because it promotes *equality of opportunity*. Moreover, the scheme is simple and shows promise for use in the design and implementation of future ESP-like GWAP systems.

References

- [1] ESP Dataset. <http://www.hcomp2009.org/Data.html>.
- [2] ESP Lite. <http://nrl.iis.sinica.edu.tw/GWAP/ESPLite/>.
- [3] Google Image Labeler. <http://images.google.com/imagelabeler/>.
- [4] GWAP. <http://www.gwap.com/gwap/>.
- [5] Herd It. <http://www.herdit.org/>.
- [6] Human Computation Dataset. <http://hcomp.iis.sinica.edu.tw/dataset/>.
- [7] Major Miner. <http://majorminer.org/>.
- [8] L. Barrington, D. Turnbull, D. O’Malley, and G. Lanckriet. Herd It: Designing A Social Game to Tag Music. In *Human Computation Workshop*, 2009.
- [9] M. Bell, S. Reeves, B. Brown, S. Sherwood, D. MacMillan, J. Ferguson, and M. Chalmers. Eyespy: Supporting navigation through play. In *ACM SIGCHI*, 2009.
- [10] P. N. Bennett, D. Maxwell, and A. Mityagin. Learning Consensus Opinion: Mining Data from a Labeling Game. In *WWW*, 2009.
- [11] J. P. Bigham, R. S. Kaminsky, R. E. Ladner, O. M. Danielsson, and G. L. Hempton. WebInSight: making web images accessible. In *ACM SIGACCESS*, 2006.

- [12] A. Bookstein. Informetric distributions, part I: Unified overview. *Journal of the American Society for Information Science*, 41(5):368–375, 1999.
- [13] S. Casey, B. Kirman, and D. Rowland. The gopher game: a social, mobile, locative game with user generated content and peer review. In *International Conference on Advances in Computer Entertainment Technology*, 2007.
- [14] L.-J. Chen, B.-C. Wang, and K.-T. Chen. The Design of Puzzle Selection Strategies for GWAP Systems. *Journal of Concurrency and Computation: Practice and Experience*, 22(7):890–908, May 2010.
- [15] L.-J. Chen, B.-C. Wang, and W.-Y. Zhu. The Design of Puzzle Selection Strategies for ESP-like GWAP Systems. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(2), 2010.
- [16] C. Gentry, Z. Ramzan, and S. Stubblebine. Secure distributed human computation. In *ACM Electronic Commerce Conference*, 2005.
- [17] L. Grant, H. Daanen, S. Benford, A. Hampshire, A. Drozd, and C. Greenhalgh. MobiMissions: the game of missions for mobile phones. In *ACM SIGGRAPH*, 2007.
- [18] C.-J. Ho, T.-H. Chang, J.-C. Lee, J. Y.-J. Hsu, and K.-T. Chen. KissKissBan: A Competitive Human Computation Game for Image Annotation. In *Human Computation Workshop*, 2009.
- [19] J. Howe. The Rise of Crowdsourcing. *WIRED Magazine*, 14(6), June 2006.
- [20] S. Jain and D. C. Parkes. A Game-Theoretic Analysis of Games with a Purpose. In *Workshop on Internet and Network Economics*, 2008.
- [21] A. Kosorukoff. Human based genetic algorithm. In *IEEE SMC*, 2001.
- [22] E. Law and L. von Ahn. Input-agreement: A new mechanism for data collection using human computation games. In *ACM SIGCHI*, 2009.
- [23] H. Lieberman, D. Smith, and A. Teeters. Common Consensus: a web-based game for collecting commonsense goals. In *ACM Workshop on Common Sense for Intelligent Interfaces*, 2007.
- [24] S. Matyas, C. Matyas, C. Schlieder, P. Kiefer, H. Mitarai, and M. Kamata. Designing Location-based Mobile Games With A Purpose: Collecting Geospatial Data with CityExplorer. In *ACM ACE*, 2008.
- [25] J. Postel. Transmission Control Protocol. IETF RFC 793, September 1981.
- [26] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision*, 77(1-3):157–173, May 2008.
- [27] P. Shenoy and D. S. Tan. Human-aided computing: utilizing implicit human processing to classify images. In *ACM SIGCHI*, 2008.
- [28] K. Siorpaes and M. Hepp. Games with a Purpose for the Semantic Web. *IEEE Intelligent Systems*, 23(3):50–60, May/June 2008.
- [29] J. Surowiecki. *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*. Doubleday, May 2004.
- [30] L. von Ahn. Games with a Purpose. *IEEE Computer*, 39(6):92–94, June 2006.
- [31] L. von Ahn and L. Dabbish. Labeling Images with a Computer Game. In *ACM SIGCHI*, 2004.
- [32] L. von Ahn and L. Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, August 2008.
- [33] L. von Ahn, S. Ginosar, M. Kedia, and M. Blum. Improving Image Search with PHETCH. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007.
- [34] L. von Ahn, S. Ginosar, M. Kedia, R. Liu, and M. Blum. Improving accessibility of the web with a computer game. In *ACM SIGCHI*, 2006.
- [35] L. von Ahn, M. Kedia, and M. Blum. Verbosity: A Game for Collecting Common-Sense Facts. In *ACM SIGCHI*, 2006.
- [36] L. von Ahn, R. Liu, and M. Blum. Peekaboom: A Game for Locating Objects in Images. In *ACM SIGCHI*, 2006.
- [37] I. Weber, S. Robertson, and M. Vojnovic. Rethinking the ESP Game. Technical report, Microsoft Research MSR-TR-2008-132, 2008.
- [38] D. H. Wilson, A. C. Long, and C. Atkeson. A context-aware recognition survey for data collection using ubiquitous sensors in the home. In *ACM SIGCHI*, 2005.