

# On the Complexity of Katamari Damacy

Gregory M. Zaverucha  
David R. Cheriton School of Computer Science  
University of Waterloo  
Waterloo ON, N2L 3G1, Canada  
gzaveruc@cs.uwaterloo.ca

September 24, 2007

## Abstract

We analyze the complexity of the popular PlayStation 2 game, Katamari Damacy. In particular, we prove that playing Katamari Damacy optimally is NP-hard and that it cannot be approximated.

## 1 Introduction

The complexity of playing games with algorithms has been of interest to computer science for a long time. An early text focusing on mathematical games by Berlekamp, Conway and Guy [4], has since grown into a four volume collection (by the same title). Much research has gone into classic puzzle games, for both one and two players. A survey of results for classic games is Demaine [7], where it is named *algorithmic combinatorial game theory*. (Not to be mistaken with *economic game theory* which deals with things like auctions.) Examples of classic games which have been studied are chess, checkers and go. Some examples of computer games with known complexity are Minesweeper [10] (a puzzle game included in Microsoft Windows), Tetris [8] and Lemmings [6]. Generally the types of games that receive attention are single player puzzle games, two player board games and mathematical games. Even the computer games which have been studied fall into these categories.

For this reason, the study of Katamari Damacy is somewhat novel since games of this genre have received little attention. Just exactly what category of game Katamari Damacy belongs to is unclear. It does share some of the properties of addictive puzzle games. It is simple to understand and start playing, but takes work to excel at. However the gameplay and style are quite different from usual puzzle games. Wikipedia refers to it as a “puzzle-action” game [2], while IGN calls it a “third-person-action” game [1]. Whatever its genre, Katamari Damacy is a break from the traditional games studied.

Many games which have stood the test of time are very difficult to play algorithmically. A common thread of these games is that they are instances of

NP-hard or NP-complete problems which are just small enough to be tractable, but challenging, for humans. Sudoku is a good example of this, since the generalized problem (with arbitrary dimensions) is NP-complete [3]. In this respect we will show that Katamari Damacy is not so different.

## 2 The Game

### 2.1 Commercial Details

We reproduce a few details from [2]. Katamari Damacy was designed by Keita Takahashi then developed and published by Namco. The game was first released for the PlayStation 2 console in Japan on March 18, 2004 and in North America in September of the same year. It has both one and two player modes (we restrict our attention to single player). Both the Computer Entertainment Rating Organization and Entertainment Software Rating Board give the game a rating which is suitable for players of all ages. The game received multiple awards for its innovative design and soundtrack.

### 2.2 Game Storyline and Description

The game's story is summarized in the game manual [9] as follows:

In a freak accident, the King of All Cosmos inadvertently annihilated all the stars. The task of fixing the King's mistake has fallen upon his son, the Prince. In order to restore the glorious starry sky, the Prince must roll around a heap of objects on Earth, gathering more and more things from the item-rich planet and sending them off into the night sky. Does the Prince have what it takes to succeed?

The heap of objects is named the *katamari* which translates roughly to "clump" from Japanese (damacy translates to "spirit") [2]. When the Prince begins a level, the katamari is small. The player controls the Prince, who pushes the katamari to make it roll around. Small objects will cling to it, and it grows in size. As the katamari grows, larger and larger objects will cling to it as well, and eventually people, elephants, buildings and trees can be collected. Each of the "Make a Star" levels has two goals (we only consider this type of level). First the katamari must reach a size specified by the King before the timer expires. If there is time to spare after reaching this size the second goal is then to make the katamari as large as possible. When the timer expires the King brings the Prince and the katamari back to space with the Royal Rainbow. If the katamari is large enough, the King creates a star from it; and if the first goal was reached quickly a comet appears in the sky as well.

Then our question is, for the Prince to build a star of size  $n$ , how long can we expect the King of All Cosmos to wait?

## 2.3 Intended Audience

We hope this paper will be enjoyed by as many fans of the game as possible. To this end, the presentation will be kept low-brow and we have made judicious use of informal arguments. Ideally this paper will be accessible to undergraduate readers who have had some exposure to complexity theory. The main section of the paper will rely on results from the common undergraduate textbook of Cormen, Leiserson, Rivest and Stein [5]. Instructors using this text for a course may therefore find this paper makes an interesting example to supplement those given in Chapters 34 and 35.

## 2.4 Formal Description

In this section we will use the terms “player” and “algorithm” synonymously, since an algorithm to play the game can be viewed as an automated player (a “computer player”).

In order to work with the game we must make some simplifying assumptions. Each of the assumptions however, will only make the game easier for players. We will then show that playing the simpler version of Katamari Damacy is NP-hard, then conclude that the actual (and more difficult) game is hard as well.

The first simplifying assumptions are that the landscape is a 2-dimensional plane, and the player’s motion within the plane is unrestricted. The latter assumption will be named the *unrestricted motion assumption*. All distances in the plane are the standard Euclidean distance. An instance of the game  $\mathcal{G}$  will be defined as a collection of “sub-games”. Each of the sub-games correspond to the the katamari’s current size, which we discretize and represent as an integer in the range  $1, \dots, s$ . The katamari starts small, so the player’s initial size is 1. If the player collects all of the objects in the game they would have size  $s$ . So an instance  $\mathcal{G}$  is divided quite naturally into the series of sub-games  $\mathcal{G}_1, \dots, \mathcal{G}_s$ .

We further assume that at sub-game  $\mathcal{G}_i$ , the only objects in the environment are those that the player can collect (i.e. objects of size  $i$  or less). Let  $\mathcal{T}_i$  be this set of objects, and represent each object by a point in the plane. Further, let  $t_i$  be the cardinality of  $\mathcal{T}_i$ .

We can now make a further assumption on the game, which we call the *perfect information assumption*. With perfect information, for  $\mathcal{G}_i$  the player is assumed to have complete knowledge of the set  $\mathcal{T}_i$ . The player completes  $\mathcal{G}_i$  by collecting all of the objects in  $\mathcal{T}_i$ . But this should be simple! Given a set of points in a plane, the player must plan a path which includes every point. Indeed if the game were this simple it would certainly not have enjoyed the success it has.

The player does not have to collect *all* of the objects at a certain size  $i$ . Once sufficiently many have been collected, the player can move on to size  $i + 1$ . To deal with this, we might formally require that the player need only collect  $k_i$  of  $t_i$  objects. Which  $k_i$  are collected is left up to the player.

The game has two goals, or criteria which are used to evaluate the player's performance (the names are inspired by the game).

1. **The comet goal:** Reach size  $x$  in minimal time. We say a player plays optimally with respect to the comet goal if the time required to reach size  $x$  is the least time possible  $T_{\min}$ .
2. **The star goal:** Grow the katamari as large as possible within the time limit. With respect to the star goal, play is optimal if the katamari is the maximum possible size  $S_{\max}$  when the timer expires. If time allows the player to pick up all of the objects in  $\mathcal{G}$ , then  $\mathcal{G}$  is played optimally if all objects are collected in the least amount of time.

### 3 Complexity

In this section we show that playing Katamari Damacy optimally with respect to either of the goals in the previous section is NP-hard. The next lemma will form the crux of this proof (Theorem 3.7).

**Lemma 3.1.** *Let  $\mathcal{P}$  be a player which plays  $\mathcal{G}$  optimally with respect to either the star goal or the comet goal. Then  $\mathcal{P}$  is able to find a path through  $k_i$  of  $t_i$  objects from  $\mathcal{T}_i$  of minimal distance (for all  $\mathcal{G}_i$ ).*

*Proof.* Recall that we assumed that the katamari rolls at constant speed. Suppose that for some  $\mathcal{G}_i$ , the path followed to collect the required  $k_i$  objects, call it  $P$ , is longer than the minimal path  $P_{\min}$ . Since the speed is constant, longer paths require more time to follow. Let  $\mathcal{P}'$  be identical to  $\mathcal{P}$  except  $\mathcal{P}'$  uses  $P_{\min}$  in  $\mathcal{G}$ . In the case of the comet goal,  $\mathcal{P}'$  finishes faster than  $\mathcal{P}$ . For the star goal, there are two possibilities. If everything in the world is picked up, then  $\mathcal{P}'$  requires less time than  $\mathcal{P}$ . If not all objects are picked up,  $\mathcal{P}'$  will be larger than  $\mathcal{P}$  since it has time to pick up more objects. In any case,  $\mathcal{P}'$  outperforms  $\mathcal{P}$ , which is a contradiction, since  $\mathcal{P}$  is optimal. Therefore, an optimal player must use shortest paths.  $\square$

At this point, most computer scientists will have recognized that playing a subgame  $\mathcal{G}_i$  optimally is closely related to the *traveling salesman problem* (TSP). We start with the definition of the TSP.

**Definition 3.2.** *The traveling salesman problem (TSP)*

**INPUT:** A graph  $G = (V, E)$  with  $n$  vertices, and an  $n \times n$  matrix  $D$  where  $D_{i,j}$  gives the distance (or cost) between vertex  $i$  and  $j$ .

**OUTPUT:** A Hamiltonian cycle through  $G$  of minimal length, i.e. a path through  $G$  which includes all  $n$  vertices, and is the shortest possible such path.

The TSP is a well known NP-complete problem. It will also be important to know it can be approximated, which we establish with the following two theorems. Recall that an  $\rho$ -approximation algorithm produces a solution which is never more than  $\rho$  times worse than the optimal solution. The parameter  $\rho$  is called the *approximation ratio*. Approximation algorithms are the subject of [5, Chapter 35]. The *triangle inequality* (in a geometric setting) essentially says that the shortest path between two points is always a straight line.

**Theorem 3.3.** *There exists a polynomial-time 2-approximation algorithm for the traveling-salesman problem with the triangle inequality.*

*Proof.* See [5, Algorithm APPROX-TSP-TOUR]. □

For more precise results about how close an approximate solution can be to optimal (for instances with the triangle inequality), see the paper of Papadimitriou and Vempala [11]. Without the triangle inequality, there is no polynomial time approximation algorithm.

**Theorem 3.4** ([5], Theorem 35.3). *If  $P \neq NP$  then for any constant  $\rho \geq 1$ , there is no polynomial-time approximation algorithm with approximation ratio  $\rho$  for the general traveling salesman problem.*

Earlier we pointed out that the player does not necessarily have to collect *all* of the objects in a particular subgame. The player can choose to roll up  $k_i$  of  $t_i$  objects in  $\mathcal{T}_i$  for  $k_i \leq t_i$  ( $k_i$  must be greater than some minimum number of objects required to make the katamari large enough to proceed to the next subgame). Once the player has decided which  $k_i$  objects to collect (using an oracle or otherwise), they are faced with the following problem.

**Definition 3.5.** *The  $k$  of  $n$  traveling salesman problem ( $(k, n)$ -TSP)*

**INPUT:** A graph  $G = (V, E)$  with  $n$  vertices, an integer  $k \leq n$ , and an  $n \times n$  matrix  $D$  where  $D_{i,j}$  gives the distance (or cost) between vertex  $i$  and  $j$ .

**OUTPUT:** A path through  $k$  vertices of  $G$  of minimal length.

The  $(k, n)$ -TSP generalizes the problem in Definition 3.2, which is the special case  $(n, n)$ -TSP. We now show that this variant of traveling salesman is no easier than the original.

**Theorem 3.6.** *The  $(k, n)$ -TSP is NP-hard.*

*Proof.* Suppose we can efficiently solve the  $(k, n)$ -TSP when  $k$  is some fixed fraction of  $n$ , say  $k = \lfloor n/d \rfloor$ . Now, given an instance of TSP of with  $n$  vertices, we can solve it in the following way. Let  $m$  be the maximum distance between any two of the original  $n$  vertices. Let  $n' = dn$  and add  $n' - n$  vertices, such that the distance between the new vertex and any of the old vertices is larger than  $m$ . Further, the distance between any two of the new vertices should also exceed  $m$ . Thinking in the 2D case, the vertices in the TSP instance we wish to solve form a “clump” and the vertices we add are “out at the horizon in

different directions”. Now we have an  $n$  of  $n'$  instance we can solve to get the optimal solution for the TSP instance. Therefore, solving the  $(k, n)$ -TSP is at least as difficult as the TSP.  $\square$

Suppose we remove the unconstrained motion assumption and that the environment is no longer modeled by a 2D plane. Then the player’s movements must follow the paths allowed by the game controls and Katamari physics. Further, the player’s motion is limited by larger objects and the terrain. Therefore, the triangle inequality no longer holds.

In absence of the triangle inequality, there is no polynomial time approximation algorithm with approximation ratio  $\rho$  for the  $(k, n)$ -TSP. By this same reasoning as in the proof of Theorem 3.6, the existence of such an algorithm would mean that the general TSP could also be approximated in polynomial time with approximation ratio  $\rho$ , which is not possible by Theorem 3.4. Summing up this section, Theorem 3.7 describes the complexity of Katamari Damacy.

**Theorem 3.7.** *Optimal Katamari Damacy is NP-hard. If  $P \neq NP$  there is no polynomial time algorithm which can play Katamari Damacy optimally. Further, for any constant  $\rho$  there is no polynomial time approximation algorithm with approximation ratio  $\rho$  for optimal Katamari Damacy.*

*Proof.* By Lemma 3.1 an optimal player must solve the  $(k, n)$ -TSP which is NP-hard by Theorem 3.6. Hence Katamari Damacy is NP-hard, and cannot be played optimally in polynomial time. An approximation algorithm for optimal Katamari Damacy would also approximate the  $(k, n)$ -TSP and TSP problems, therefore optimal Katamari Damacy cannot be approximated in polynomial time either.  $\square$

## 4 Conclusion

This paper has shown that optimal Katamari Damacy is NP-hard by reduction from a variant of the traveling salesman problem. One could also ask whether optimal Katamari Damacy is NP-complete. The difficulty of optimally choosing the optimal value  $k$ , and then choosing  $k$  objects to collect would also have to be dealt with. Algorithmically playing the game in two player mode would also make for interesting future work.

**Acknowledgments** The author thanks the following people for reviewing an earlier draft of this paper: Chris Hamilton, Kevin Henry, Glenn Hickey, Abram Hindle and Aniket Kate.

## References

- [1] IGN: Katamari Damacy. Accessed April 2007.
- [2] Katamari Damacy – Wikipedia, the free encyclopedia. Accessed April 2007.

- [3] L. Aaronson. Sudoku science. *IEEE Spectrum*, 43(2):16–17, February 2006.
- [4] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways for your Mathematical Plays*. Academic Press, London, 1982.
- [5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. McGraw-Hill, Boston, 2nd edition, 2001.
- [6] G. Cormode. The Hardness of the Lemmings Game, or “Oh no, more NP-completeness proofs”. Technical Report 2004-11, Center For Discrete Mathematics and Computer Science, Rutgers University, May 2004.
- [7] E. D. Demaine. Playing games with algorithms: Algorithmic combinatorial game theory. In *Mathematical Foundations of Computer Science*, Lecture Notes in Computer Science, pages 18–34. Springer-Verlag, 2001.
- [8] E. D. Demaine, S. Hohenberger, and D. Liben-Nowell. Tetris is hard, even to approximate. In *Computing and Combinatorics*, Lecture Notes in Computer Science, pages 351–363. Springer-Verlag, 2003.
- [9] Namco Bandai Games Inc. Katamari Damacy instruction booklet, 2004. Distributed with each copy of the game.
- [10] R. Kaye. Minesweeper is NP-complete. *Mathematical Intelligencer*, 22(2), 2000.
- [11] Christos H. Papadimitriou and Santosh Vempala. On the approximability of the traveling salesman problem. *Combinatorica*, 26(1):101–120, 2006.