Boulder Dash is NP–hard

Marzio De Biasi
marziodebiasi [at] gmail [dot] com

December 2011
Version 0.01: "... now the difficult part: is it NP?"

Abstract

Boulder Dash is a videogame created by Peter Liepa and Chris Gray in 1983 and released for many personal computer and console systems under license from First Star Software. Its concept is simple: the main character must dig through caves, collect diamonds, avoid falling stones and other nasties, and finally reach the exit within a time limit. In this report we show that the decision problem *"Is an $n \times n$ Boulder Dash level solvable?"* is NP–hard. The constructive proof is based on a simple gadget that allows us to transform the hamiltonian cycle problem on a 3-connected cubic planar graph to a Boulder Dash level.

1 Introduction

The videogame *Boulder Dash* [8] was created by Peter Liepa and Chris Gray in 1984 and was originally released for the Atari 8-bit system in 1984 under license from First Start Software, which still owns the rights to the game. It was a great success, so it was soon ported on other personal computers (ZX Spectrum, C64,...) and console systems (ColecoVision, NES,...); moreover it gave origin to many sequels and clones. The most famous clones are *Emerald Mine* by Voler Wertich, released for the Amiga personal computer, and the various versions of *Diamonds Caves* by Peter Elzner, released for the Amiga, MS-DOS and Windows.

1.1 Boulder Dash rules and objects

The game is subdivided into *levels*; each level is a $n \times m$ grid. Each cell of the grid can be one of the following objects:

Rockford : the main character controlled by the player. Rockford can move in one of the four directions: left, right, up, down, one cell at a time.

Space : space is just an empty cell. Rockford can freely move on space cells.

Dirt : dirt can serve for blocking and/or suspending other objects such rocks or diamonds. Rockford can freely move on a dirt cell, but he clears it and the cell becomes a space when he leaves it.

Wall : walls are unmovable and undestructable objects; Rockford cannot move on them.

Rock : rocks are undestructable and Rockford cannot move on them, but he can push a single rock horizontally if there is a space beside it. If the dirt is removed from beneath, the rock falls until it reaches a solid ground again. A rock can also roll off other rocks or walls if there is enough space. If Rockford is hit by a falling rock, he dies and the level is restarted.

Diamond : diamonds are the items that Rockford must collect in order to complete a level.

Exit door : at the beginning of a level, the exit door is closed. When Rockford collects all the diamonds, the exit door opens and Rockford must reach it to complete the level.

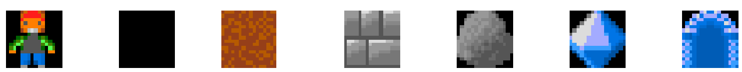

Figure 1: Game objects: Rockford, Space, Dirt, Wall, Diamond, Exit door.

The original game has other objects: *Fireflies*, *Butterflies*, *Amoebas*, *Expanding Walls*, *Magic Walls*, *Steel Walls*; but we will not use them. Furthermore in the original game, there is also a time limit for each level; but we will assume that Rockford has *unlimited time* to collect the diamonds and reach the exit door.

2 NP–hardness of a puzzle game

In complexity theory a *decision problem* is a problem expressed in some formal system, for which the desired answer is yes or no. The complexity class P contains all the decision problems that can be solved by a deterministic Turing machine in polynomial time. The complexity class NP contains all the decision problems whose solutions can be verified by a deterministic Turing machine in polynomial time. The question if $P = NP$? is perhaps the major still-open problem in computer science.

The *NP–complete* class of problems contains all decision problems such that they are in NP and every other problem in NP is *Karp reducible* to them in polynomial time: a problem A is Karp-reducible to a problem B if there is a polynomial time algorithm that given as input an istance I_A of problem A produces as output an instance I_B of problem B and the answer for I_A is yes iif the answer for I_B is yes. For a complete introduction to NP–completeness see the book of Garey and Johnson: *"Computers and Intractability"* [3].

Many of the puzzle games that people play are interesting because they require intuition and logic reasoning to be solved. Most of them can be mathematical modeled and the difficulty of solving them is deeply tight to the complexity of the corresponding decision problem.

For example the following puzzle games have been proved to be NP–complete [5]: Sudoku, Kakuro, Instant insanity, Light up, Lemmings, Clickomania, Mastermind, FreeCell.

The *NP–hard* class of problems contains all decision problems that are "at least as hard as the hardest problems in NP": a problem A is NP–hard if and only if there is as NP–complete problem B that is polynomial time Turing-reducible to A.

Some puzzles are even harder and are $PSPACE$–complete, the complexity class of decision problems solvable by a Turing machine in polynomial space and such that any other problem in PSPACE can be reduced to them. For example Atomix [4], Rush Hour [1] and Sokoban [6] have been proved to be $PSPACE$–complete.

3 Boulder Dash is NP–hard

We will prove that the decision problem *"Is an $n \times n$ Boulder Dash level solvable?"* is NP–hard. The constructive proof is based on a polynomial reduction from the NP–complete problem *hamiltonian cycle on a 3-connected cubic planar graph (3PHC)* [2].

A graph is *k-connected* if there does not exist a set of $k - 1$ vertices whose removal disconnects the graph. A *planar* graph is a graph that can be drawn on te plane in such a way that its edges intersects only at their vertices. A *cubic* graph is a graph in which all vertices have degree three. A *3-connected cubic planar* graph is at the same time 3-connected cubic and planar.

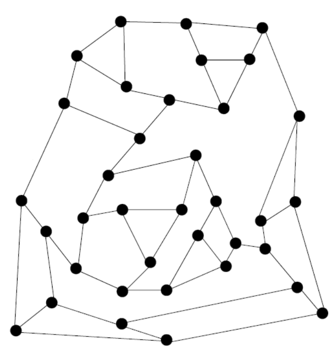

Figure 2: A 3-connected planar cubic graph.

A *Hamiltonian cycle* or *Hamiltonian circuit* in an undirected graph, is a sequence of edges that forms a closed loop (the last vertex is also the starting one) and visits each vertex exactly once. The decision problem of determining whether a Hamiltonian Cycle exists in a given undirected graph is NP–complete. The problem stays NP–complete even if the input graph is 3-connected planar cubic (3PHC) [2].

A planar graph can be drawn on the plane in many different ways; a drawing of a graph in which vertices and bends are located at grid points of an integer grid is called a *grid drawing*. We will use a particular grid drawing called *orthogonal drawing*: each edge is drawn on the grid as a chain of horizontal and vertical line segments. An orthogonal drawing for 3-connected cubic planar graphs can be found very efficiently in *linear time* [7]. Furthermore, the size of the resulting othogonal drawing for a 3-connected cubic planar graph is polynomial in the number of the nodes: if n is the number of nodes of the original graph, and $W \times H$ is the size of the resulting graph, we have $W \leq \frac{n}{2}$ and $H \leq \frac{n}{2}$ [7].

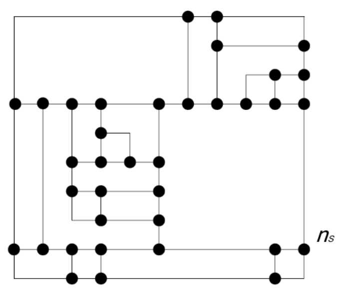

Figure 3: Othogonal drawing of the graph in Fig.2.

We recall that we will use only the following game objects: *Rockford*, *Wall*, *Dirt*, *Space*, *Rock*, *Diamond*, *Exit door*; we will also assume that the game has *no time limit*.

Consider the following two *"gadgets"* that are possible valid parts of a Boulder Dash level:

Node–gadget : the Node–gadget is an area made by solid walls, that has three exits (left, up, and down). It contains a diamond, three rocks and some dirt. The rocks are sustained by dirt, and it is easy to see that if Rockford moves under one of them the rock falls and blocks the passage. Therefore, in order to pick the diamond, Rockford must enter a Node–gadget through one of the three passages, and exit from another one. He must pick the diamond because it is placed in the crossing of the three passages. The third passage remains free, but if Rockford uses it, he will be imprisoned in the gate.

Start–gadget : the Start–gadget is similar to the Node–gadget but contains no diamonds; it defines the Rockford starting position and the Exit door

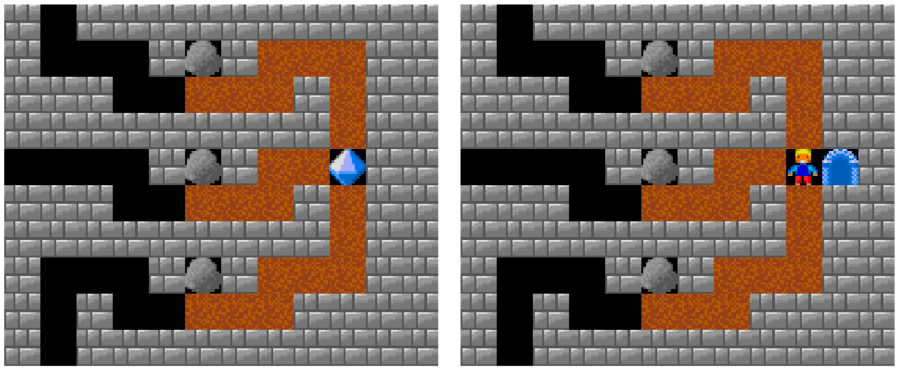

Figure 4: *Node*–gadget and *Start*–gadget.

position. Rockford must leave the Start–gadget in order to get the diamonds (leaving two free passages); but can re-enter it only when all the diamonds have been collected. If it tries to use the remaining two free passages to reach other diamonds, the passages become blocked and the Exit door cannot be reached anymore.

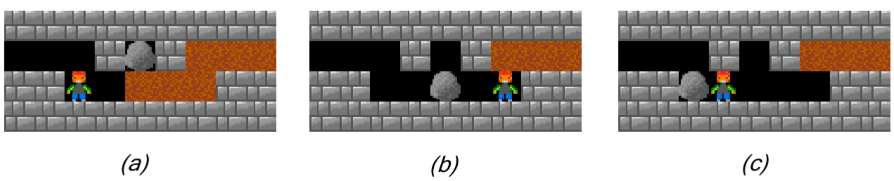

Figure 5: Rockford enters a passage of the gadget *(a)*; the rock falls *(b)*; the passage is blocked both from left to right and right to left *(c)*.

Both gadgets can be horizontally mirrored (in order to bring the middle passage on the right) without affecting their properties. We can also extend the upper and lower exits and bring them on the right of the gadget. In such a manner we have four Node-gadgets (and four Start-gadgets) which shapes trace out the four possible "T" connections on a grid (Fig.6).

The gadgets can be linked together with horizontal, vertical, and bends *tunnels* made of walls (Fig.7).

Now, given an orthogonal grid drawing of a planar cubic graph $G = (N, V)$, we can build a Boulder Dash level L that exactly traces out the shape of G using one Start–gadget and $|N| - 1$ Node–gadgets linked together with horizontal, vertical and bend tunnels. The Start–gadget can be placed in correspondence of an arbitrary node n_S of N. The size of each gadget is 19×19, so the upper bound for the size $W_L \times H_L$ of level L is $W_L \leq \frac{19 \times |N|}{2}$ and $H_L \leq \frac{19 \times |N|}{2}$

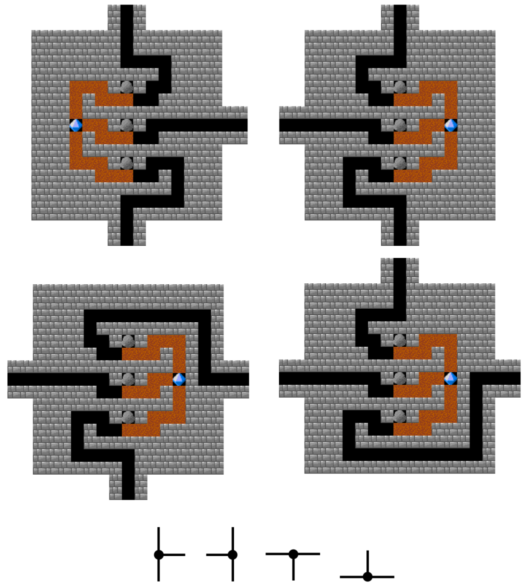

Figure 6: Gadgets of size 19×19 rearranged to simulate four T grid connections.

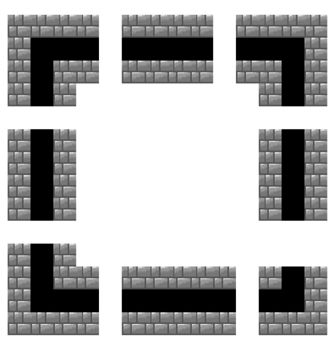

Figure 7: Tunnels made with walls can be used to link the gadgets together.

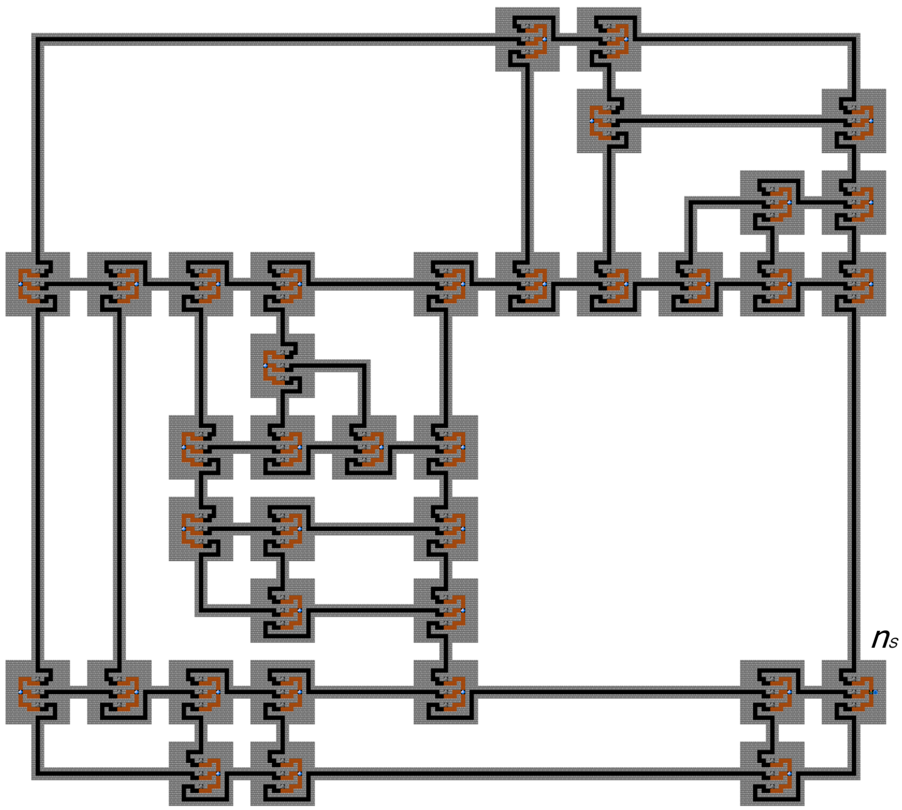

Figure 8: The Boulder Dash level built from the orthogonal drawing of Fig.3.

We observe that the actions of Rockford, which is placed in the Start–gadget, are limited to:

1. choose one of the three available exits of the Start–gadget (two stays opened);
2. move along the tunnel to another Node–gadget (but he cannot go back);
3. enter the Node–gadget (and thus block that exit);
4. pick the diamond and choose one of the two remaining exits (the third stays opened);
5. move along the tunnel to another Node–gadget;
6. repeat steps 2–4 until he gets blocked on a Node or collects all the Diamonds and reaches the Start-gadget again (where the Exit door is placed).

Furthermore:

- if Rockford re-enters an already visited Node–gadget (through the third free exit), he gets blocked inside;
- if Rockford re-enters the Start–gadget (through one of the two free exits) before collecting all the Diamonds, he can exit again, but he will never be able to re-enter it and use the Exit door (and thus will never complete the level);

Lemma 3.1. *Rockford can complete level L* iif *the graph G has a Hamiltonian cycle.*

Proof. If the graph G has a Hamiltonian cycle, then, by construction, the same cycle is a valid path that Rockford can follow to solve the level L: the cycle will lead him to every node allowing him to pick all the diamonds; each node will be traversed only once so he will always found a free entrance and a free exit. Finally the cycle will bring him back to the Start-gadget and to the opened Exit door.

Conversely suppose that the level L has a solution. Rockford must visit every Node-gadget because they contains the Diamonds; furthermore for the properties showed above, each Node–gadget can be visited only once and the Start–gadget must be visited again only at the end (when the Exit door is open); so Rockford must follow a cycle to solve the level and the path followed by Rockford determines an Hamiltonian cycle in the original graph G.

□

Theorem 3.2. *Boulder Dash is NP–hard.*

Proof. We proved that given a 3-connected cubic planar graph G, we can build a corresponding Builder Dash level L in polynomial (linear) time which has polynomial size in the number of nodes of G and is solvable iif G has a Hamiltonian cycle. This reduction from an NP–complete problem to the problem of determining if a level is solvable permits us to conclude that Boulder Dash is NP-hard.

□

4 Conclusions

Play Boulder Dash and have fun!

References

[1] Flake and Baum. Rush hour is PSPACE-complete, or "why you should generously tip parking lot attendants". *TCS: Theoretical Computer Science*, 270, 2002.

[2] Garey, Johnson, and Tarjan. The planar hamiltonian circuit problem is NP-complete. *SICOMP: SIAM Journal on Computing*, 5, 1976.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability.* Freeman, San Francisco, 1979.

[4] M. Holzer and S. Schwoon. Assembling molecules in ATOMIX is hard. *Theoretical Computer Science*, 303(3):447–462, 2004.

[5] Graham Kendall, Andrew J. Parkes, and Kristian Spoerer. A survey of NP-complete puzzles. *ICGA Journal*, 31(1):13–34, 2008.

[6] Sokoban Is Pspace-complete, Joseph C. Culberson, and Joseph C. Culberson. Sokoban is pspace-complete, 1997.

[7] Md. Saidur Rahman. Efficient algorithms for drawing planar graphs, 1999.

[8] Wikipedia. Boulder dash — wikipedia, the free encyclopedia, 2011. [Online; accessed 4-December-2011].