# Solving the 15-Puzzle

Chris Calabro

June 14, 2005

**Abstract**

The $n \times n$ puzzle game is played on a matrix of numbered tiles with 1 tile missing to allow tiles to shift. The goal is to order the tiles by a sequence of shifts.

We provide a $O(n^2)$-time algorithm to decide when an initial configuration of the $n \times n$ puzzle game is solvable. We also provide an algorithm solving the game in $O(n^3)$ moves and show that this is asymptotically optimal.

## 1 The game

We now define the $m \times n$ puzzle game. Let $U = [m] \times [n]$. We impose the taxi cab metric on $U$: for $p = (i, j), q = (i', j') \in U, d(p, q) = |i - i'| + |j - j'|$. A special element $B \in U$, usually $(m, n)$, will be designated as the "blank".

A *game state* is a bijection $\sigma : U \to U$. $\sigma(p) = q$ means that in position $p$ is the label $q$. A *transposition* permutation is a bijection $\tau : U \to U$ such that $\exists p, q \in U, p \neq q\ \tau(p) = q, \tau(q) = p, \forall r \in U - \{p, q\}\ \tau(r) = r$. $\tau$ represents interchanging the elements in positions $p, q$. A *legal move* from state $\sigma$ is a transposition permutation $\tau$ interchanging $p, q$ where $\sigma(p) = B$ and $d(p, q) = 1$. The state of the game after application of $\tau$ is $\sigma \circ \tau$.

A move $\tau$ interchanging $p = (i, j), q = (i', j')$ where the state just before the move is $\sigma$ and $\sigma(p) = B$ is a *left* move iff $i = i' - 1$, a *right* move iff $i = i' + 1$, an *up* move iff $j = j' - 1$, a *down* move iff $j = j' + 1$.

The initial state of the game is some permutation $\sigma : U \to U$. The (one and only) player makes a sequence of legal moves and wins iff the last state is the identity permutation.

The $4 \times 4$ game with $B = (4, 4)$ is most common and is often called the 15-puzzle. It is not known when this game was first sold but at least one version was sold by the Embossing Co. in 1865.

## 2 A necessary condition for solvability

Notice that if the game is solvable and $\sigma(B) = B$, then in a winning sequence of moves, the number of left moves must equal the number of right moves and the number of up moves must equal the number of down moves, simply because the blank ends up at the same position as where it started. So the number of transpositions must be even. So the initial permutation must be even. We generalize this idea below.

**Lemma 1.** *The $m \times n$ puzzle game with initial state $\sigma$, where the blank starts in position $p$, is solvable only if the sign of $\sigma$ is the same as that of $d(p, B)$. I.e. only if ($\sigma$ is even iff $d(p, B)$ is even).*

The following algorithm computes the sign of a permutation $a[1], \dots, a[n]$ of $1, \dots, n$ in time $O(n)$, which is optimal.

```
sign($a[1, \dots, n]$)
  $s \leftarrow 0$
  $i \leftarrow 1$
  while $i \leq n$
    if $i \neq a[i]$
      swap($a, i, a[i]$)
      $s \leftarrow 1 - s$
    else
      $i++$
  return $s$
```

To see the correctness of this, notice that after the $k$th iteration of the loop, $s$ is the congruence class modulo 2 of the number of transpositions used so far to modify $a$ and all elements before position $i$ are in the correct position and at least $k$ elements are in the correct position.

So we can decide solvability of the $m \times n$ game in time $O(mn)$.

# 3 A sufficient condition for solvability

We claim that the condition in lemma 1 is sufficient as well. (provided $m, n \geq 2$)

**Lemma 2.** *Let $2 \leq m \leq n$. The $m \times n$ puzzle game with initial state $\sigma$, where the blank starts in position $p$, is solvable if the sign of $\sigma$ is the same as that of $d(p, B)$.*

*Proof.* We describe an algorithm solving it. We will suppose for now that $B = (m, n)$ and generalize later.

Suppose $(m, n) = (2, 2)$. Through a sequence of $d(p, B)$ moves, we can move the blank to position $B$. Let $\sigma$ be the state after this is done. Then $\sigma$ is even. By applying the 4 moves right, down, left, up, we can cyclically rotate the 3 non-blank labels. So all 3 even permutations with the blank in position $B$ are reachable, including the identity.

Now suppose $n \geq 3$. We order 1 row at a time, starting from the top and working left to right. Suppose that at some point we have ordered $i - 1$ rows and the first $j-1$ elements of row $i$ and now we want to put $(i,j)$ into position $(i,j)$ without disturbing the elements in earlier positions. Also suppose that $i \leq m-2, j \leq n-1$.

If $(i,j)$ is already in position, then we are done. But otherwise, choose an "L" shaped set of positions that is 2 positions thick and not containing any positions that should be left undisturbed. This can be done since $m, n \geq 2$ and $i \leq m-2$ and $j \leq n-1$. Move the blank around the position labeled $(i,j)$ in a spiral fashion inside this "L" to propagate it to the correct position.

E.g. suppose the "L" is a $3 \times 2$ matrix, $(i,j) = (1,1)$ is currently at position $(3,1)$, the blank is at position $(2,1)$. Then use up, left, down, down, right, up. This takes $O(m+n)$ moves.

To get the last element of the row into position, move $(i,j)$ into the position just below position $(i,j)$ and move the blank into the position just below that. Then use down, down, right, up, left, up, right, down, down, left, up. Note that this temporarily disturbs position $(i,j-1)$. It only moves the labels in a $3 \times 2$ submatrix.

This strategy can be used to order the last 2 rows as well by rotating our perspective by 90 degrees.

At the end we will have a $2 \times 2$ submatrix containing the blank and the remaining labels. Since we have preserved the invariant that the sign of the permutation matches that of the distance of the blank from position $B$, this $2 \times 2$ subproblem can be solved in place as well.

Now suppose $B \neq (m,n)$. We can use the same basic strategy as before but order the rows furthest from $B$ first and work inwards. At the end we will have a $2 \times 2$ subproblem just as before, but now it will be somewhere in the middle. □

Notice that the above algorithm uses $O((m+n)mn)$ moves.

## 4 Lower bound on the number of moves

To get a non-blank label from position $p$ to $q$ takes $\geq d(p,q)$ moves and each such move moves *only* that non-blank label. So the number of moves required to solve the game is at least $\sum_{p \in U - \{B\}} d(\sigma(p), p)$. If $\sigma$ is chosen uniformly at random from all solvable permutations, then $\forall p \in U \ E(d(\sigma(p), p)) \in \Theta(m+n)$. By linearity of expectation, the expected number of moves required is at least $\Omega((m+n)mn))$.

In particular, there is an initial configuration requiring at least $\Omega((m+n)mn)$ moves, but the average case statement is stronger. Since our algorithm achieves this bound, it is optimal, up to a constant factor.

## References

[1] Jaap Scherphuis, Jaap's Puzzle Page, http://www.geocities.com/jaapsch/puzzles/fifteen.htm