EXPECTED RUNNING TIME OF 2-SAT RANDOM LOCAL SEARCH

William Wu
wwu@ocf.berkeley.edu
http://www.ocf.berkeley.edu/~wwu/

2002 September 23

1 Background Math: Solving Linear Recurrences

(Material in this section taken from Chapter 5.2 of Discrete Math and Its Applications, by Kenneth H. Rosen.)

Theorem 1. *Let $c_1, c_2, \ldots, c_k$ be real numbers. Suppose the characteristic equation*

$$r^k - c_1 r^{k-1} - \ldots - c_k = 0$$

has t distinct roots $r_1, r_2, \ldots, r_t$ with multiplicities $m_1, m_2, \ldots, m_t$, respectively, so that $m_i \geq 1$ for i = 1, 2, ..., t and $m_1 + m_2 + \ldots + m_t = k$. Then a sequence $\{a_n\}$ is a solution of the recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}$$

if and only if

$$\begin{aligned} a_n &= (\alpha_{1,0} + \alpha_{1,1} n + \ldots + \alpha_{1,m_1-1} n^{m_1-1}) r_1^n \\ &\quad + (\alpha_{2,0} + \alpha_{2,1} n + \ldots + \alpha_{2,m_2-1} n^{m_2-1}) r_2^n \\ &\quad + \ \ldots \ + (\alpha_{t,0} + \alpha_{t,1} n + \ldots + \alpha_{t,m_t-1} n^{m_t-1}) r_t^n \end{aligned}$$

Theorem 2. *If $\{a_n^p\}$ is a particular solution of the nonhomogeneous linear recurrence relation with constant coefficients*

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} + F(n)$$

then every solution is of the form $\{a_n^p + a_n^h\}$, where $\{a_n^h\}$ is a solution of the associated homogeneous recurrence relation

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k}.$$

Theorem 3. *Suppose that* $\{a_n\}$ *satisfies the linear nonhomogeneous recurrence relation*

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} + F(n)$$

where $c_1, c_2, \ldots, c_k$ *are real numbers and*

$$F(n) = (b_t n^t + b_{t-1} n^{t-1} + \ldots + b_1 n + b_0) s^n,$$

where $b_0, b_1, \ldots, b_t$ *and* s *are real numbers. When* s *is not a root of the characteristic equation of the associated linear homogeneous recurrence relation, there is a particular solution of the form*

$$(p_t n^t + p_{t-1} n^{t-1} + \ldots + p_1 n + p_0) s^n$$

When s *is a root of this characteristic equation and its multiplicity is* m*, there is a particular solution of the form*

$$n^m (p_t n^t + p_{t-1} n^{t-1} + \ldots + p_1 n + p_0) s^n$$

2 Randomized 2-SAT Recurrence

Here is a randomized local search algorithm for finding a satisfying assignment to a given 2-SAT expression:

1. Choose initial truth assignment equally at random
2. While there exists an unsatisfied clause:
   a. Choose some unsatisfied clause C equally at random
   b. Pick one of the two variables in C equally at random, and flip its truth value

This algorithm is not guaranteed to find a satisfying assignment. However, after you run it for a large number of iterations (what's a large number? I'll figure that out later, but probably $100n^2$), the probability that it finds a satisfying assignment comes close to 100%, if a satisfying assignment exists.

What is the expected number of iterations this algorithm takes to complete?

Note that we can think of this algorithm's behavior as akin to a random walk. After each iteration, the number of variables in our truth assignment which match the correct truth assignment will either increment or decrement by one. This is the key observation.

Let us declare the following variables and functions:

n = number of boolean variables in the 2-SAT expression

S(k) = expected number of iterations till all n variables in the algorithm's truth assignment match the correct satisfying assignment, given that k variables already match

Thus if we can calculate S(0) – the expected number of iterations given that no variables are initially correct – we will have a worst-case analysis.

I will argue that the function S(k) is given by the following recurrence:

$$S(k) = \begin{cases} \frac{1}{2}(S(k+1)+1) + \frac{1}{2}(S(k-1)+1) & \forall k \in \{\ k \in \mathcal{Z} \mid 0 < k < n\} \\ 0 & k = n \\ S(1)+1 & k = 0 \end{cases} \tag{1}$$

Explanation: If k variables match right now, then after inverting one variable's assignment, either k+1 or k-1 variables will match the correct assignment. In the worst case, both cases will occur with probability 0.5.[1]And in both cases, I add 1 to the conditional expectation, to count the iteration just consumed.

As for the base cases, S(n) = 0 because if all n variables have assignments which match the correct assignment, the algorithm ends, and you don't need any more iterations. S(0) = S(1) + 1 because if 0 variable assignments match the solution, then inverting the assignment of a variable can only result in 1 variable assignment matching the solution; it doesn't make sense to say that -1 variables match.

Now we solve for S(0). The closed-form of this recurrence will be the sum of two parts, a homogeneous solution $S(k)^h$ and a particular solution $S(k)^p$.

STEP 1: Homogeneous solution

Ignoring the base cases for now, note that we can rewrite 1) as $S(k) = \frac{1}{2}S(k+1)+\frac{1}{2}S(k-1)+1$. Now clearly, the homogeneous version of (1) is

$$S(k) = \frac{1}{2}S(k+1) + \frac{1}{2}S(k-1).$$

We now write a characteristic equation for this homogeneous equation. Assume there exists a constant r such that

$$r^k = \frac{1}{2}r^{k+1} + \frac{1}{2}r^{k-1}.$$

Dividing both sides by r^{k-1} yields:

$$r = \frac{1}{2}r^2 + \frac{1}{2}$$

[1]We assume that in every iteration of the algorithm, when an unsatisfied 2-SAT clause is chosen and one of the two variable assignments is inverted, there is always a 0.5 probability of moving toward the correct solution, and a 0.5 probability of moving backwards. However in reality, if both variable assignments in the chosen unsatisfied clause are currently incorrect, then the algorithm will move forwards with probability 1. So this analysis gives a worst-case bound that may not be tight.

Solving for r using the quadratic formula yields:

$$r = 1 \qquad \text{with multiplicity 2 (double root).}$$

Thus $S(k)^h = (\alpha_1 k + \alpha_2)r^n = (\alpha_1 k + \alpha_2)1^k = (\alpha_1 k + \alpha_2)$, where α_1 and α_2 are constants. Note that we choose a solution of this form because the multiplicity is two.

STEP 2: Particular solution

The nonhomogeneous component of (1) is the function F(k) = 1. F(k) is a polynomial of degree zero, and shares a mode with the homogeneous solution. Assume there exists a particular solution of the form $S(k)^p = k^2(p_0)(1)^k = k^2 p_0$, where p_0 is a constant. Substituting $S(k)^p$ into (1) yields:

$$\begin{aligned}
p_0 k^2 &= 0.5p_0(k+1)^2 + 0.5p_0(k-1)^2 + 1 \\
&= 0.5p_0(k^2 + 2n + 1) + 0.5p_0(k^2 - 2k + 1) + 1 \\
&= p_0 k^2 + p_0 + 1 \\
0 &= p_0 + 1 \\
p_0 &= -1
\end{aligned}$$

Thus $S(k)^p = -k^2$.

STEP 3: Combine homogeneous and particular solutions

Adding the homogeneous and particular solutions yields:

$$S(k) = S(k)^h + S(k)^p = (\alpha_1 k + \alpha_2) - k^2$$

From the initial conditions listed in (1):

$$\begin{aligned}
S(0) &= S(1) + 1 \\
(\alpha_1 * 0 + \alpha_2) - 0^2 &= ((\alpha_1 * 1 + \alpha_2) - 1^2) + 1 \\
\alpha_2 &= (\alpha_1 + \alpha_2) \\
\alpha_1 &= 0
\end{aligned}$$

$$\begin{aligned}
S(n) &= 0 \\
0 &= (\alpha_1 n + \alpha_2) - n^2 \\
&= \alpha_2 - n^2 \\
\alpha_2 &= n^2
\end{aligned}$$

Substituting $\alpha_1 = 0$ and $\alpha_2 = n^2$ into the expression for S(0):

$$\begin{aligned}
S(0) &= (\alpha_1 * 0 + \alpha_2) - 0^2 \\
&= n^2
\end{aligned}$$

Thus the expected number of iterations is n^2, and the algorithm runs in polynomial time.