Normalisation of Fitness

CS454 AI-Based Software Engineering

Shin Yoo

Normalisation

- “[with obj.] Mathematics multiply (a series, function or item of data) by a factor that makes the norm or some associated quantity such as an integral equal to a desired value (usually 1). “
- Seems innocuous, but the choice of normalisation method can affect optimisation itself.

Normalisation

- Linear: requires bounds.
- Widely used:
- Suggested by Arcuri 2010:

$$\omega_0(x) = 1 - \alpha^{-x}, \text{ where } \alpha > 1$$

$$\omega_1(x) = \frac{x}{x + \beta}, \text{ where } \beta > 0$$

Requirements

- Should preserve the partial order of the raw values.

$$\omega : \mathbb{R}^+ \to [0, 1]$$

$$x_i < x_j \Leftrightarrow \omega(x_i) < \omega(x_j)$$

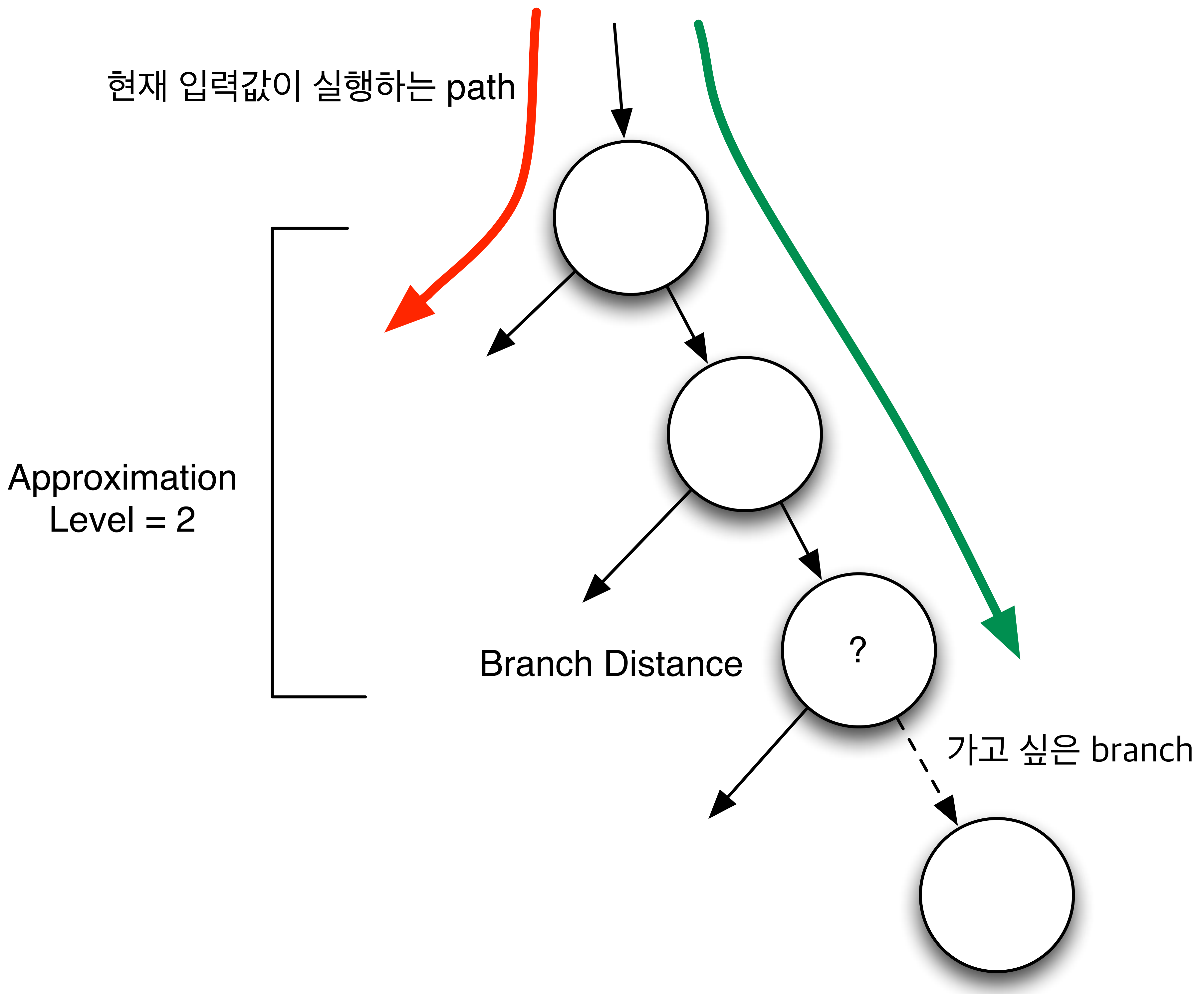
$$x_i = x_j \Leftrightarrow \omega(x_i) = \omega(x_j)$$

$$\omega_0(x+\epsilon) = 1 - \alpha^{-(x+\epsilon)} = 1 - (\alpha^{-x} \cdot \frac{1}{\alpha^{\epsilon}}) > 1 - \alpha^{-x} = \omega_0(x)$$

$$\begin{aligned}
\omega_1(x+\epsilon) &= \frac{x+\epsilon}{x+\epsilon+\beta} \\
&= \frac{x+\epsilon}{x+\epsilon+\beta} + \frac{\beta}{x+\epsilon+\beta} - \frac{\beta}{x+\epsilon+\beta} \\
&= 1 - \frac{\beta}{x+\epsilon+\beta} \\
&> 1 - \frac{\beta}{x+\beta} = \frac{x}{x+\beta} = \omega_1(x)
\end{aligned}$$

Test Data Generation

- Branch coverage Fitness function = [approach level] + ω([branch distance])
- Current candidate input diverges from the wanted path that leads to the target:
  - Approach level: # of nesting levels to penetrate in order to reach the target
  - Branch distance: distance in the target predicate between current state and the wanted outcome (true/false)



Branch Distance

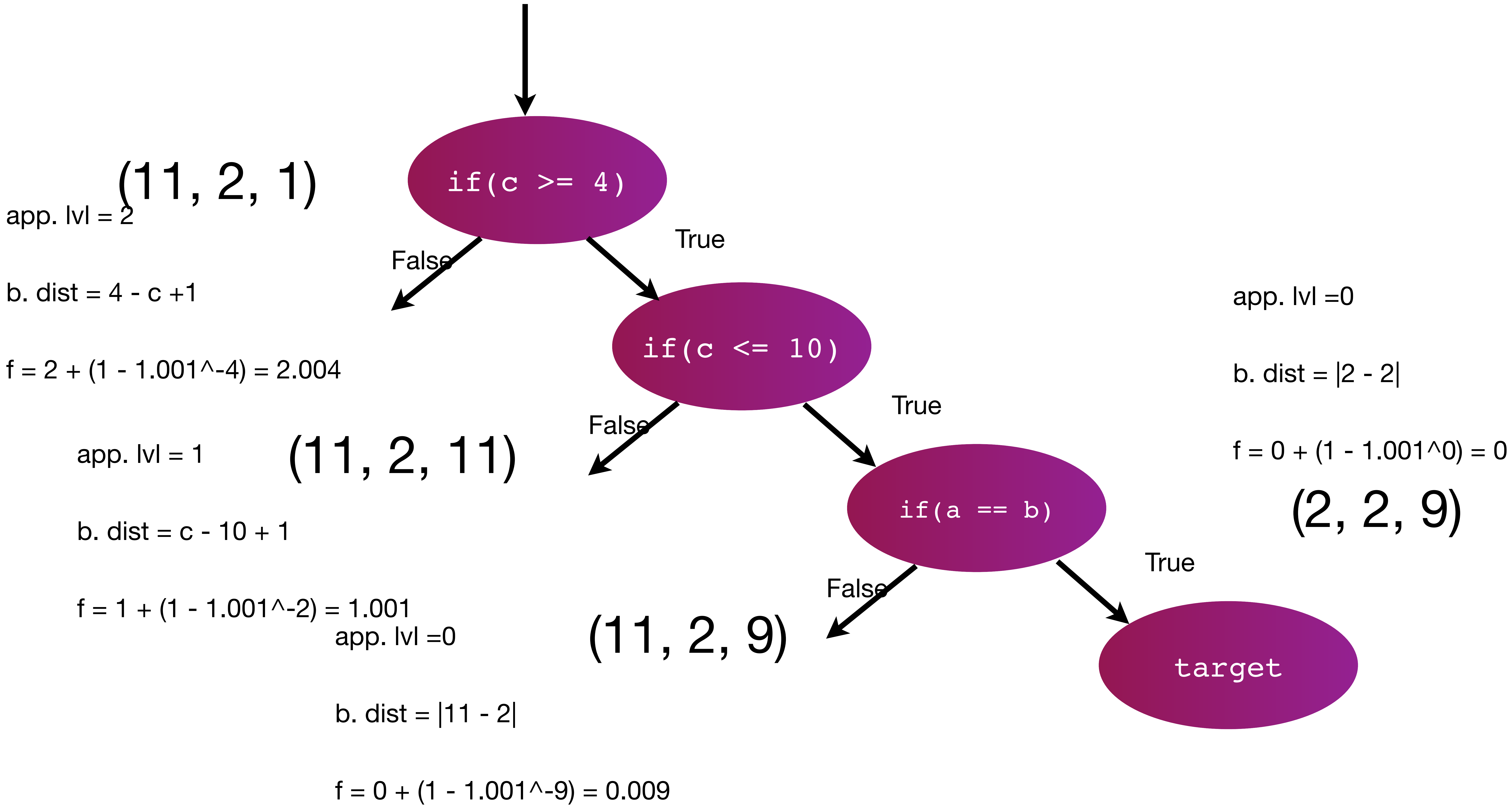
- But predicates are boolean. What do you mean “distance”?
- Satisfying x == y: convert to b = |x - y|, and minimise b. When b becomes 0, x becomes equal to y.
- Satisfying y >= x: convert to b = x - y + K, and minimise.When b becomes 0, y is greater than x by K.
- Normalisation is necessary: branch distance cannot be bigger than approximation level (which is a “count” measure).
  - Literature prefers $\omega_0(b) = 1 - 1.001^{(-b)}$

Branch Distance

Predicate	f	minimise until..
a > b	b - a + K	f < 0
a >= b	b - a + K	f <= 0
a < b	a - b + K	f < 0
a <= b	a - b + K	f <= 0
a == b	$\lvert a - b \rvert$	f == 0
a != b	$-\lvert a - b \rvert$	f < 0

B. Korel, “Automated software test data generation,” IEEE Trans. Softw. Eng., vol. 16, pp. 870–879, August 1990.

Fitness Function



Test input (a, b, c), K = 1

Suppose we use SA

Under ω_0

Probability of accepting a worse neighbour ρ_n from ρ at temperature T:

$$P_{\rho_n} = e^{-\frac{f(\rho_n) - f(\rho)}{T}}$$

Let approximation level $\mathcal{A}(\rho)$, be branch distance be $\delta(\rho)$, and $\gamma = e^{1\frac{1}{T}}$ for convenience. Assume ρ and ρ_n share the same approximation level, which is common during optimisation.

$$\begin{aligned}
P_{\rho_n}^{\omega_0} &= \gamma^{(\mathcal{A}(\rho_n)+\omega_0(\delta(\rho_n))-\mathcal{A}(\rho)+\omega_0(\delta(\rho)))} \\
&= \gamma^{(\omega_0(\delta(\rho)+\epsilon)-\omega_0(\delta(\rho)))} \\
&= \gamma^{1-\alpha^{-(\delta(\rho)+\epsilon)}-(1-\alpha^{-\delta(\rho)})} \\
&= \gamma^{\alpha^{-\delta(\rho)}(1-\alpha^{-\epsilon})}
\end{aligned}$$

Branch distance of the current solution as an exponential effect (i.e. $\alpha^{-\delta(\rho)}$).

Suppose we use SA

Under ω_1

$$
\begin{aligned}
P_{\rho_n}^{\omega_1} &= \gamma^{(\mathcal{A}(\rho_n)+\omega_0(\delta(\rho_n))-\mathcal{A}(\rho)+\omega_0(\delta(\rho)))} \\
&= \gamma^{(\omega_0(\delta(\rho)+\epsilon)-\omega_0(\delta(\rho)))} \\
&= \gamma^{\frac{\delta(\rho)+\epsilon}{\delta(\rho)+\epsilon+\beta}-\frac{\delta(\rho)}{\delta(\rho)+\beta}} \\
&= \gamma^{\frac{\beta\epsilon}{(\delta(\rho)+\beta)(\delta(\rho)+\epsilon+\beta)}}
\end{aligned}
$$

Branch distance of the current solution has a polynomial effect (i.e. $\approx \delta(\rho)^{-2}$).

Direct impact on Runtime

Figure 4. Toy program.

- The choice of normalisation function has a side-effect on the difference of energy level when deciding whether to accept a sub-optimal solution.
- ω_0 takes longer! The exponential impact apparently is too generous(!) - see the next slide.

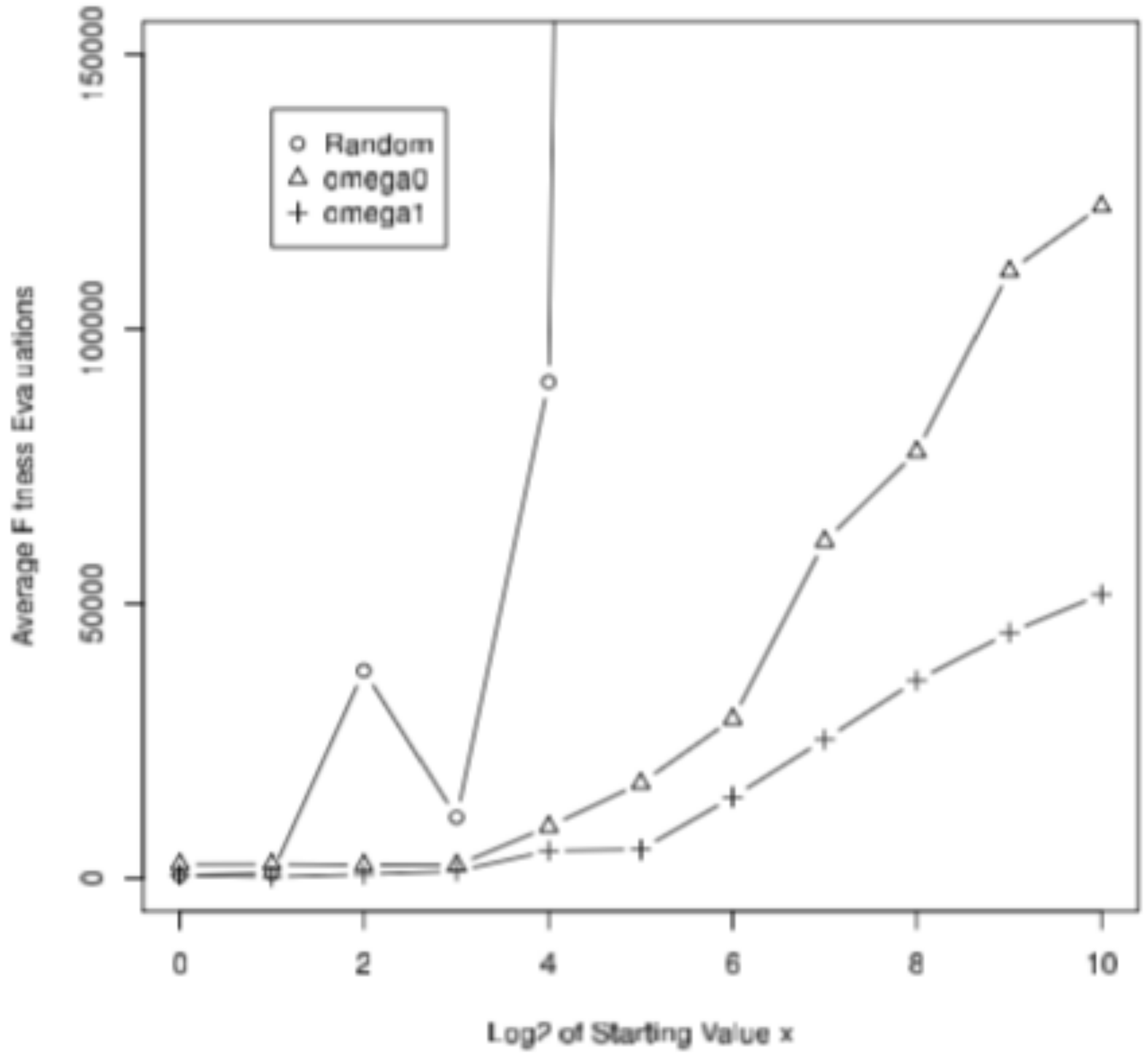


Figure 5. Average number of fitness evaluations to find input $x = 0$.

A. Arcuri. It does matter how you normalise the branch distance in search based software testing. In Software Testing, Verification and Validation (ICST), 2010 Third International Conference on, pages 205–214, April 2010.

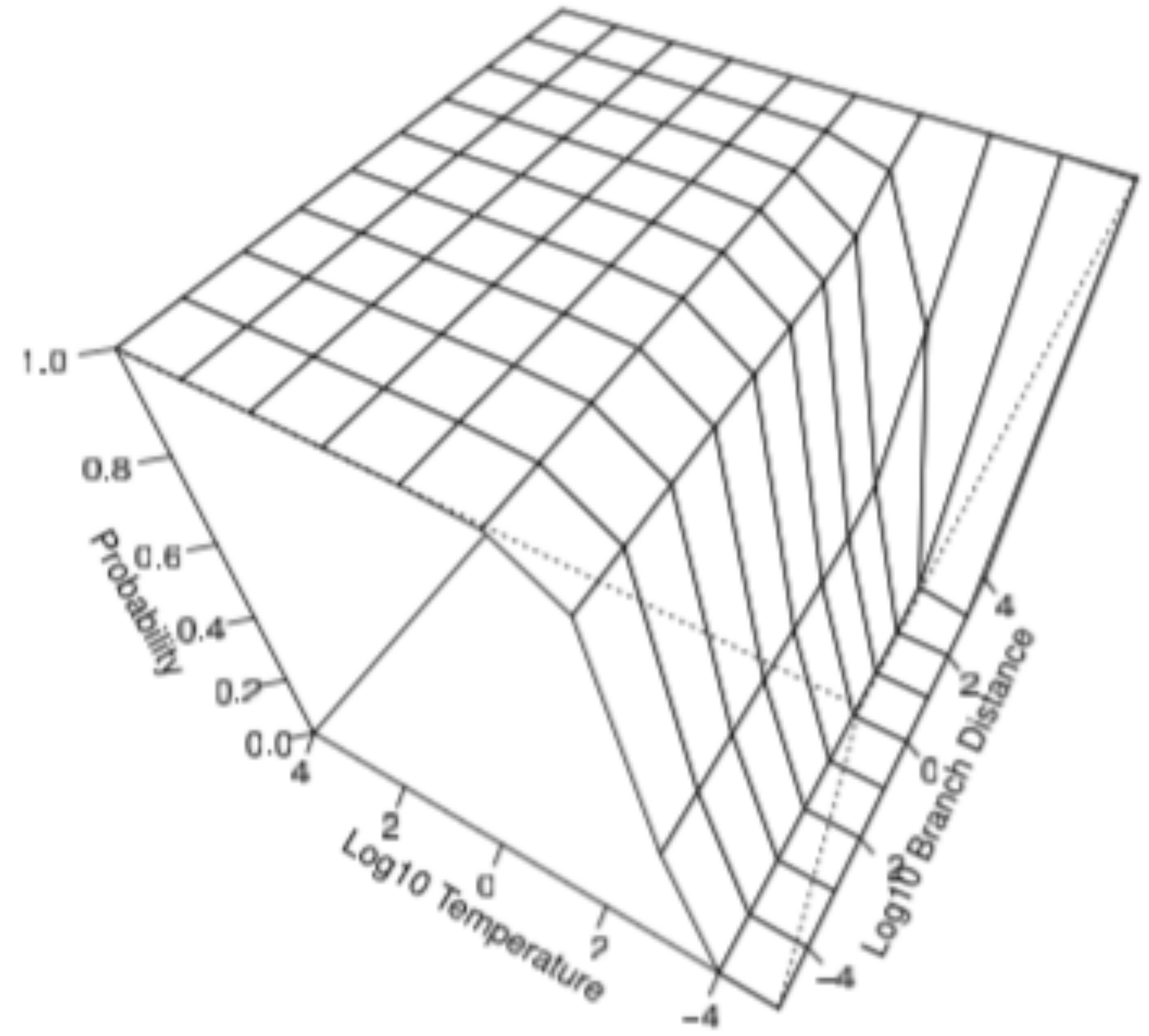


Figure 2. Probability of accepting a worse solution when ω_0 is used.

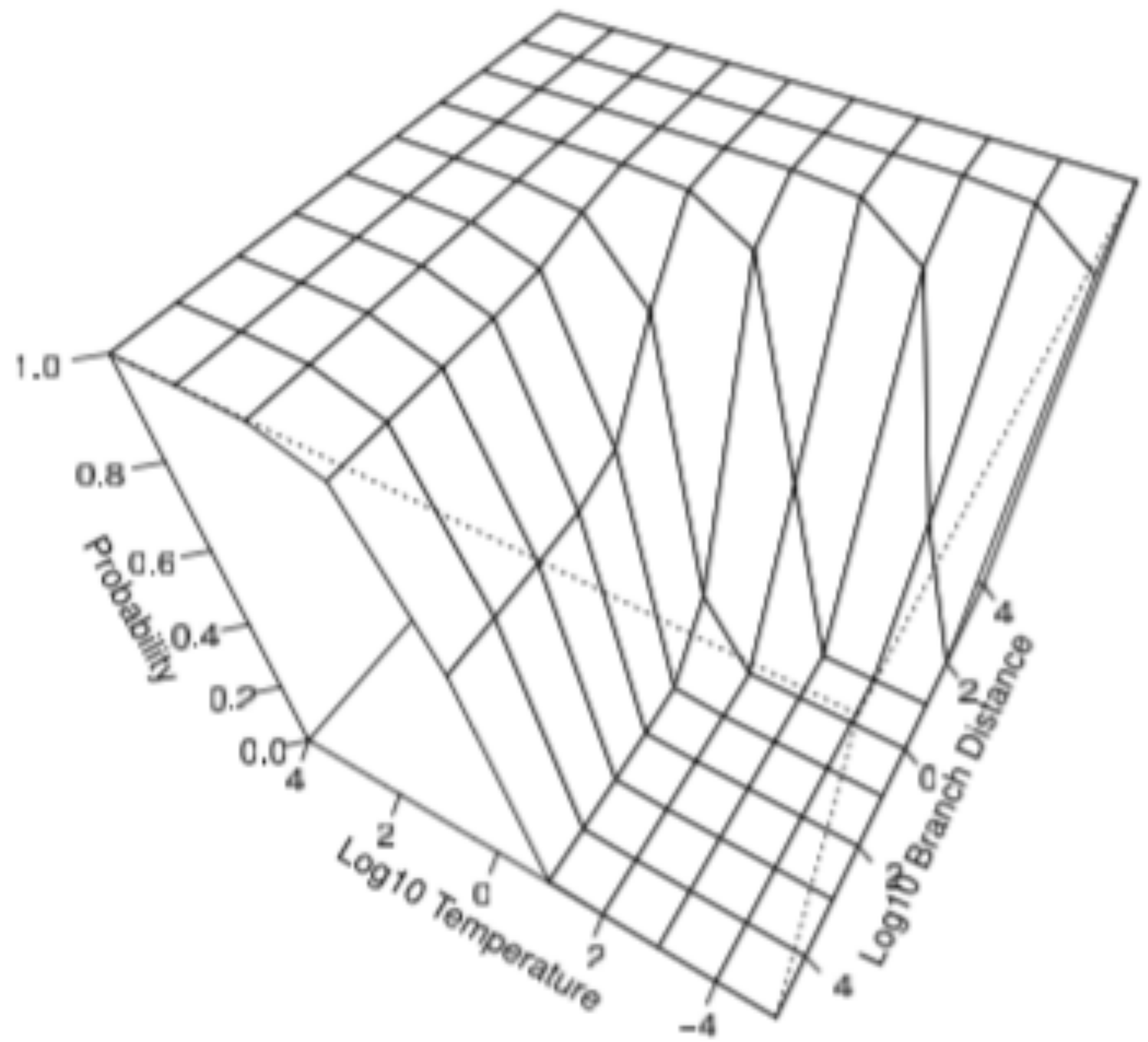


Figure 3. Probability of accepting a worse solution when ω_1 is used.

A. Arcuri. It does matter how you normalise the branch distance in search based software testing. In Software Testing, Verification and Validation (ICST), 2010 Third International Conference on, pages 205–214, April 2010.