BDI-Learning Discussion Paper #6: A Robust & Scalable Confidence Measure

Dhirendra Singh
dhirendra.singh@rmit.edu.au

March 17, 2010

1 Calculating Plan Confidence

Previously we have used the notion of coverage [2] and structural complexity [3] to capture the ongoing exploration of the goal-plan structure. The idea is that the extent of exploration of the structure in some way relates to how confident we are in the resulting decision tree. Both approaches depend on the calculation of the number of *choices* below a plan in the goal-plan hierarchy.

The main benefit of these coverage-based approaches is that they give a continuous confidence measure that is useful in fine-tuning exploration of the goal-plan structure. However the downside is that it is difficult to calculate (1) the full set of paths (specially when parameterised and recursive goals are considered); and (2) which path combinations beginning in a failed trace should be "counted" as covered since some combinations never eventuate.

Compare this to *stability* [1, 2], a boolean measure of confidence that is not very useful for guiding exploration, nonetheless is effectively equivalent to coverage [2] only without the complication of paths calculations.

In this section, we explore a confidence measure that combines the merits of both the coverage-based and stability-based measures. The idea is that confidence for a plan P in a given world w may be defined as *the ratio of the number of stable plans below P to the total number of plans tried below P, starting in world w*. In other words, our confidence gradually increases as plans below P start to become stable, and is maximum when every plan that has been tried below P is considered stable.

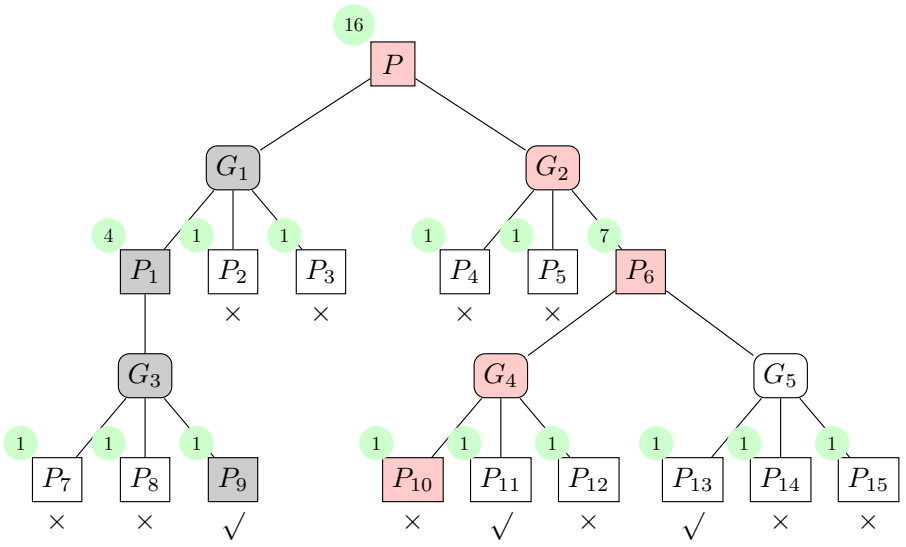

Figure 1: An example goal-plan hierarchy.

Let us consider the example goal-plan structure of Figure 1, and see how the confidence calculation might work for the top-level plan P. Let's say the first execution results in the failed trace $\lambda = G[P : w] \cdot G1[P_1 : w] \cdot G3[P_7 : w]$. The confidence in P is then:

$$confidence(P, w) = \frac{stable\ plans\ in\ [P, P_1, P_7]}{total\ plans\ in\ [P, P_1, P_7]} = \frac{0}{3} \tag{1}$$

Let's say the second execution results in the failed trace $\lambda = G[P : w] \cdot G1[P_2 : w]$. The confidence at this step is then:

$$confidence(P, w) = \frac{stable\ plans\ in\ [P, P_1, P_2, P_7]}{total\ plans\ in\ [P, P_1, P_2, P_7]} = \frac{0}{4} \tag{2}$$

Now, let's say the next four executions fail in P_2, P_8, P_2 (that becomes stable), P_4 (note that for this to happen P_9 must have been selected first). At step six then, the confidence is given by:

$$confidence(P, w) = \frac{stable\ plans\ in\ [P, P_1, P_2, P_4, P_7, P_8, P_9]}{total\ plans\ in\ [P, P_1, P_2, P_4, P_7, P_8, P_9]} = \frac{1}{7} \tag{3}$$

And so on.

In Figure 1, the number against each plan node $P*$ indicates the maximum number of sub-plans that will eventually be considered at that node for full confidence. Algorithm 1 describes this confidence calculation for a given active execution trace λ. Here η is used to propagate the full list of sub-plans in the trace up to the parent. $StablePlan$ is a boolean function to check if the given plan P_n is stable r not. $Choices$ holds the full list of all past sub-plans executed along with their stability status.

Algorithm 1: $UpdateConfidence(\lambda, \eta, k, \epsilon)$

Data: $\lambda = G_0[P_0 : w_0] \cdot \ldots \cdot G_n[P_n : w_n]$; $\eta = [P, b] \cdot \ldots \cdot [P_r, b_r]$; $k \geq 0$; $\epsilon \geq 0$
Result: Updates confidence for plans in λ

```
choices(P_n, w_n) = choices(P_n, w_n) ∪ P_n ∪ η;
confidence(P_n, w_n) += |choices(P_n, w_n) == true| / |choices(P_n, w_n)|;
if |λ| > 1 then
    λ' = G_0[P_0 : w_0] · ... · G_{n-1}[P_{n-1} : w_{n-1}];
    η' += [P_n, StablePlan(P_n, k, ε)];
    UpdateConfidence(λ', η', k, ε);
```

The first benefit of this approach is that the confidence grows gradually, in a manner similar to coverage, and may be used to similarly guide exploration. The second benefit is that we do not have to concern ourselves with possibilities (paths) that may never eventuate, as we do with coverage. Say if plan P_9 was to also fail in world w. In that case goal G_2 would never be posted and all it's sub-plans would never be executed. For coverage-based calculations this is a problem since we consider the full gamut of possibilities, and ignoring such combinations would mean that full coverage would never be achieved. With this new measure however, since we only deal with past occurrences then we are still guaranteed to converge to full confidence as long as the set of previously tried plans are executed sufficient times to become stable. The third benefit is that the method scales well as it does not depend on the complexity of the domain, but only on the execution history.

One limitation of this (and our previous approaches) is that the confidence measure is a per-world concept while the decision tree that it is applied to encapsulates knowledge from multiple worlds. This means that if a new world is witnessed, then even though the decision tree might generalise based on past worlds, the confidence measure will always be zero resulting is the plan selection weight to default to 0.5. One way to improve this situation for unseen worlds would be to use an average confidence value from all previously seen worlds. This would allow generalisations to be used, but with some reservation based on confidence.

To Resolve: P_9 is not part of the active execution trace ending in P_4 so will be missed in Equation 3.

2 Calculating Domain Confidence

Here, instead of determining our confidence in the plan (based on exploration of the structure), the aim is to build a confidence measure to capture the complexity of the domain (i.e number of worlds). The motivating idea for the solution is that a confidence measure may be constructed around the rate at which new worlds are being witnessed by a plan. During early exploration it is expected that the majority of worlds that a plan is selected for will be unique, therefore this rate is high and our confidence is low. Over time as exploration continues, the plan would get selected in all possible worlds and the rate of new worlds would taper off towards zero. Intuitively, our confidence over this period would increase to it's maximum.

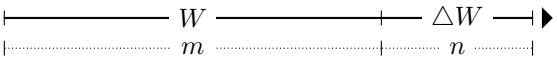

Figure 2: Passage of worlds with time for a given plan.

In inverse terms, confidence would increase with the rate at which old worlds are being witnessed. Figure 2 illustrates the passage of time from the point of view of a given plan. Here W is the set of all worlds witnessed in the first m experiences and $\triangle W$ is the set of worlds witnessed in the last n experiences. Then $|W \cap \triangle W|$ gives us the number of old worlds seen in the last n experiences. For a given plan T, Equation 4 gives the ratio that defines our domain confidence measure $0 \leq \kappa_T \leq 1.0$ that is guaranteed to converge to 1.0 if $n <= m$ and as long as all worlds are eventually witnessed. The value n may be computed as $n = a \cdot m$ where $0.0 < a \leq 1.0$ is a user-specified option.

$$\kappa_T = \frac{|W \cap \triangle W|}{n} \tag{4}$$

3 Putting It all Together: Calculating Plan Selection Weight

For plan T, the plan confidence $c_T(w)$ and domain confidence κ_T may then be used to calculate a final plan selection weight $\Omega(w)$ as given by Equation 5. Here $p_T(w)$ is the probability of success in world w given by the decision tree of plan T.

$$\Omega'_T(w) = 0.5 + [c_T(w) * \kappa_T * (p_T(w) - 0.5)]\,. \tag{5}$$

References

[1] S. Airiau, L. Padgham, S. Sardina, and S. Sen. Enhancing Adaptation in BDI Agents Using Learning Techniques. In *International Journal of Agent Technologies and Systems*, 2009.

[2] D. Singh, S. Sardina, L. Padgham, S. Airiau. Learning Context Conditions for BDI Plan Selection. In *Proceedings of Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2010.

[3] D. Singh, S. Sardina, L. Padgham. Extending BDI Plan Selection to Incorporate Learning from Experience. Submission under review for the Special Issue on Hybrid Control of Autonomous Systems (HYCAS), Journal of Robotics and Autonomous Systems, 2010.