Simulation experiments for hide-and-seek with different seeker distribution update strategies

Tirtharaj Dash

March, 2020

1 Setup

We perform some simulation experiments for Hide-and-Seek with three different **seeker distribution update strategies**:

(1) **No update:** No update of the seeker distribution (leads to hide-and-seek with replacement results).

(2) **Uniform update:** Open a box, distribute its probability mass to every other unopened boxes, make its probability 0 (hide-and-seek without replacement).

(3) **Hot-cold update:** The seeker updates its probability distribution based on whether it opened a cold box or a hot box. A cold box is a box for which the performance of a box ($perf$) is less than the cold threshold (θ_c) and a hot box is a box with performance greater than a hot threshold (θ_h). We devise the following procedure for update:

   1. Open a box i
   2. If $perf(i) \geq \theta_h$: distribute its probability mass to all the unopened boxes in its neighbors.
   3. If $perf(i) \leq \theta_c$: distribute its probability mass to all the unopened boxes except its neighbors.
   4. If none of 2 or 3: distribute its probability mass to all the unopened boxes.
   5. Repeat 1–4 until the hider is found.

(4) **Localised-sampling update:** The idea is that if an opened box is a "good" box then its neighbors may also be good. So, we propose the following update strategy:

   1. Open a box i
   2. If the hider is not found in that box, **but it is a good box (i.e.** $perf \geq \theta_h$**)**, open its unopened neighbors starightaway and check if the hider is found there.

3. If hider is not found, then distribute the probability mass of all the boxes (the sampled box and its neighbors) to all unopened boxes equally.
4. If not 2., then distribute the mass of the opened box i to all unopened boxes equally.

The seeker update strategy (3) requires that the hider distribution falls into some continuity assumption. That is: the probability mass of a neighborhood of a box are in monotonic relationship to the probability of that box. This is a realistic demand and clauses which are related with each other are monotonic in their performance in some fashion. We construct such a hider distribution (H) with the following code[1].

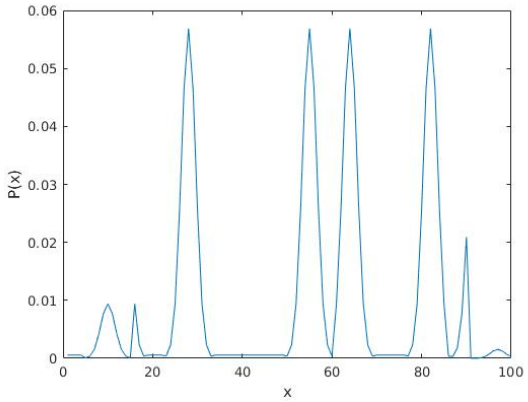

Figure 1: Sample hider distibution

2 Experiments – Base

Parameter setting The experiments are performed for number of boxes $n = \{1000, 2000, 3000\}$. The maximum hiding trials is set at 1000. We call it a **failure**, if the hider is not found within n searches by using the seeker distribution. The neighborhood size (nbd) is varied as {1,2,3}. For all the experiments reported here, we define performance of a box by: $perf(i) = \frac{h_i}{\max(h_1,\ldots,h_n)}$, where $H = \{h_1, \ldots, h_n\}$ is the hider distribution. The thresholds are fixed at $\theta_h = 0.80$ and $\theta_c = 0.4$. The proportion of boxes that have high probability mass (spikes in H) is fixed at 10%. For the seeker update stragegy described in 4 requires a neighborhood size ($lnbd$). This is varied as {1,2,3}.

Results The mean and standard deviations of misses are calculated only for successful runs i.e. the hider was found by the seeker within maximum of n look-ups. Otherwise, it was treated as a failure and this result was not included for statistics. Below, we report results for each seeker update strategies.

[1] `https://github.com/tirtharajdash/multimodalGaussianDistro`

choiceUpdS	SuccessRate	mean(misses)	sd(misses)
		$n = 1000$	
1	0.622	439.738	279.218
2	1.000	499.379	279.060
3 ($nbd = 1$)	1.000	486.041	271.251
3 ($nbd = 2$)	1.000	458.027	269.987
3 ($nbd = 3$)	1.000	479.270	279.110
4 ($lnbd = 1$)	1.000	521.648	281.406
4 ($lnbd = 2$)	1.000	559.455	283.613
4 ($lnbd = 3$)	1.000	569.803	290.259
		$n = 2000$	
1	0.635	859.123	571.302
2	1.000	1022.204	595.486
3 ($nbd = 1$)	1.000	963.457	550.133
3 ($nbd = 2$)	1.000	961.946	551.028
3 ($nbd = 3$)	1.000	968.335	564.515
4 ($lnbd = 1$)	1.000	1053.850	578.713
4 ($lnbd = 2$)	1.000	1118.901	560.618
4 ($lnbd = 3$)	1.000	1105.028	559.592
		$n = 3000$	
1	0.652	1303.462	854.176
2	1.000	1473.717	865.756
3 ($nbd = 1$)	1.000	1436.214	801.100
3 ($nbd = 2$)	1.000	1396.195	840.066
3 ($nbd = 3$)	1.000	1464.990	840.695
4 ($lnbd = 1$)	1.000	1611.752	872.266
4 ($lnbd = 2$)	1.000	1632.951	855.368
4 ($lnbd = 3$)	1.000	1725.939	853.418

Figure 2: Average number of misses: H is a multimodal distribution with 10% spikes, and S is updated using three different update strategies (1: no update, 2: without replacement, 3: update using θ_h and θ_c using three different neighborhoods in {1,2,3}, 4: local sampling based update with local neighborhood in {1,2,3}).

Interpretation 1 The results suggest that the threshold based seeker update strategy (3):

- Reduces the average number of misses in comparison with search without replacement for which the expected number of misses is $\frac{n-1}{2}$ (i.e. choiceUpdS= 2 in tables).
- For the seeker updat type (3), best neighborhood size is found to be 2.
- Increasing the neighborhood size of a box increases the average number of misses

in almost all the cases.

Interpretation 2 The localised sampling based seeker update strategy described in 4 has following effects on average number of misses.

- It is worse than to seeker update strategy 3.
- Bigger neighborhood sizes have adverse effect on the avergae number of misses. This could be due to the fact that we are starting with a uniform neighborhood and we don't have much idea about the true hider distribution yet.

2.1 Experiments – Resource-bounded Localised-sampling Update

The localised sampling update strategy described in 4 contributes a lot of costs towards opening neighbors of a box when the box is a good box. Why does this happen? In the present case, as soon as the seeker finds a good box, it becomes greedy and opens its neighbors. This happens everytime it finds a good box. Opening the neighbors incurs costs, which gets added to the total number of misses. Now, we can restrict this number of neighbor opening to some $MAX\_LS \in (0, 1)$ (maximum local sampling), where MAX_LS could be a function of n. We define this constant as follows:

$$MAX\_LS = k * n \tag{1}$$

That is neighbor opening is allowed k percentage of n times. We call this strategy as 'Resource-bounded localised sampling update'.

Observation: It is easy to verify that this is a **more general** case of the seeker update strategy 3 and a **more specific** case of seeker update strategy 4. That is:

- $MAX\_LS = 0$ results in hot-cold update in 3.
- $MAX\_LS = \infty$ results in localised-sampling update in 4.

Results We conduct experiments with $k = \{0.01, 0.05, 0.10\}$. The comparison is against the results obtained for the localised-sampling seeker update (4). The neighborhood of a box is fixed at 1. Refer Fig.3.

Interpretation Here are the observations.

- As k increases, the cost increases. This is expected.
- In all cases, $k = 0.10$ results in 4.
- The results are worse than that of the hot-cold update strategy.

choiceUpdS	SuccessRate	mean(misses)	sd(misses)
	$n = 1000$		
3 ($nbd = 1, k = 0$)	1.000	486.041	271.251
4 ($lnbd = 1, k = \infty$)	1.000	521.648	281.406
4 ($lnbd = 1, k = 0.01$)	1.000	512.810	283.890
4 ($lnbd = 1, k = 0.05$)	1.000	528.908	290.191
4 ($lnbd = 1, k = 0.10$),	1.000	521.648	281.406
	$n = 2000$		
3 ($nbd = 1, k = 0$)	1.000	963.457	550.133
4 ($lnbd = 1, k = \infty$)	1.000	1053.850	578.713
4 ($lnbd = 1, k = 0.01$)	1.000	986.059	573.569
4 ($lnbd = 1, k = 0.05$)	1.000	1031.794	572.295
4 ($lnbd = 1, k = 0.10$),	1.000	1053.850	578.713
	$n = 3000$		
3 ($nbd = 1, k = 0$)	1.000	1436.214	801.100
4 ($lnbd = 1, k = \infty$)	1.000	1611.752	872.266
4 ($lnbd = 1, k = 0.01$)	1.000	1594.132	842.846
4 ($lnbd = 1, k = 0.05$)	1.000	1560.329	858.586
4 ($lnbd = 1, k = 0.10$),	1.000	1611.752	872.266

Figure 3: Resource-bounded localised-sampling seeker update.

3 Conclusions

The seeker update strategy as described in 3 is the **best** update strategy, when the hider is not known, but a structural relationship among the boxes is known. The performance associated with a box can only be known after opening the box. This kind of hider structure space is applicable to ILP hypothesis space, where the space forms a lattice of clauses and each clause is associated with a performance (e.g. Laplacian smoothed accuracy: $\frac{P+1}{P+N+2}$, where P and N are the number of positive and negative instances respectively).

Our next set of experiments will be testing the applicability of this new seeker update strategy for ILP hypothesis obtained for real-world problems as described in a separate report.