Task Similarity Measures for Transfer in Reinforcement Learning Task Libraries

James L. Carroll
Department of Computer Science
Brigham Young University
Provo, Ut 84602
E-mail: james@jlcarroll.net

Kevin Seppi
Department of Computer Science
Brigham Young University
Provo, Ut 84602
E-mail: kseppi@cs.byu.edu

***Abstract*— Recent research in task transfer and task clustering has necessitated the need for task similarity measures in reinforcement learning. Determining task similarity is necessary for selective transfer where only information from relevant tasks and portions of a task are transferred. Which task similarity measure to use is not immediately obvious. It can be shown that no single task similarity measure is uniformly superior. The optimal task similarity measure is dependent upon the task transfer method being employed. We define similarity in terms of tasks, and propose several possible task similarity measures, d_T, d_P, d_Q, and d_R which are based on the transfer time, policy overlap, Q-values, and reward structure respectively. We evaluate their performance in three separate experimental situations.**

I. Introduction

In an extensive reinforcement learning task library there may be many tasks that are related to the target task, but there may also be many tasks that are unrelated to the target task. Since the task transfer algorithms currently in use are all extremely sensitive to the nature and amount of similarity that is present between the source and the target tasks [1][2][3][4], a method for quantifying the similarity of two tasks is needed.

Unfortunately, task "similarity" is an ill defined term. What does it mean for two tasks to be "similar" and for another two tasks to "not be similar?" How can levels of similarity be quantified? Quantification can be even more complex because tasks can be similar in several ways.

One possible method for defining similarity would be in terms of content, meaning that similar tasks share specific features. However, it is unclear how much weight should be given to each shared feature. Even if this problem could be overcome, humans find similarity between tasks in more complex ways, for example, through analogy or metaphor. Analogies are "those problems that share a similar deep structure but not necessarily specific content" [5]. This means that tasks that have no features in common can still be considered similar.

We define task similarity in this paper with respect to a given transfer technique, where the level of similarity under a given transfer technique is an approximation to the "advantage" gained by using one task to speed the learning of another task. We posit that any general quantification of task similarity outside of this definition is meaningless. Furthermore, this definition of task similarity places the need to identify "deep structure" on the shoulders of the task transfer mechanism since if such a technique is developed, the advantage gained by using that technique can then be quantified.

II. Desirable Properties of a Task Similarity Measure

One of the standard uses for task similarity measures is to select a source task that can be used when learning a given target task. This is an important step since transfer from similar tasks can greatly speed the learning of the target task, while transfer from an unrelated source task can greatly degrade performance.

Therefore, one possible task similarity measure would be to actually learn a target task given a source task, and somehow measure the "advantage" gained by using the source task and call that the measure of similarity between the two tasks. This can capture the deep, analogical, or metaphorical similarity between the two tasks so long as the transfer technique is capable of utilizing such similarity. In some ways this could be the best method for measuring similarity between two tasks.

However, in other ways this technique is not helpful. Although it produces a measure of similarity, it only produces such a measure after the transfer experiment has been run. If the point is to use the task similarity measure to choose a task to use in transfer, then this task similarity measure produces a measure of similarity after it is too late to use such a measure.

It should be noted that such a measure of similarity need not be a "metric" or a "measure" in the mathematical sense of those terms, nor should it be. The process of shaping [6], for example, is based on the idea that it is faster to learn an intermediary task, and then use that task to aid the learning of a more complex task than it is to learn the more complex task from scratch. Thus shaping depends on the fact that the triangle inequality does not hold for task similarity when task similarity measures the advantage of using one task to aid the learning of another; nor is it clear that the properties of symmetry or identity should necessarily hold.

We formally define the term "task similarity measure" and its inverse "task distance measure $d(i, j)$," as a heuristic function that has the following desirable properties:

1) The task similarity measure should provide an approximation to the amount of learning improvement that we would get when using the source task to learn the target task under a given transfer technique.
2) If $d(l, i) > d(l, k)$ then we would hope that using task k to aid in the learning of task l, would provide a better bias for learning task l than using task i. Thus the task similarity measure should provide an approximation to a partial ordering for the similarity between task i and the rest of the tasks in L (where L is a task library or set of tasks).
3) The task similarity measure should be computable without actually running the transfer experiment. In other words it should be able to produce an approximate partial ordering before task l has been thoroughly learned, while the information could still be of use to aid in the learning of l. The ordering can be refined as the experiment runs, but the measure should provide a useful approximation before the learning is complete.

III. There is No Best Task Similarity Measure

Having a "best" measure of similarity is like having a "best" inductive bias. We would like to be able to say with some certainty exactly how similar two things are. But given the endless possibilities for analogies and metaphors, such a measure is impossible.

It can be easily shown that there exists two transfer techniques that will cause different source tasks to be more "advantageous" when learning a given target task, (as was demonstrated in [7]). It can also be shown that there exists a single source task and two target tasks and two transfer techniques, such that one transfer technique will be more effective when learning one target task, while the other will be more effective for learning the other target task.

Intuitively this is because the transfer technique imposes a bias on the target task. Since there is no best bias for learning all target tasks (the "no free lunch theorem") there is no best transfer technique to employ for all target tasks. Further, one source task may be more useful under one transfer technique than another. Therefore, if similarity is defined as the expected usefulness of using one source task to speed the learning of another target task, there can be no "best" task similarity measure apart from the transfer technique employed.

This doesn't mean that task similarity measures are not useful in task libraries. However, this does mean that we can not directly talk about how similar two tasks are and that task similarity must be defined relative to some transfer mechanism.

IV. Proposed Task Similarity Measures

We propose several task similarity measures, d_T, d_P, d_Q, and d_R.

d_T is the technique already described above, where the transfer experiment is actually run, and the "advantage" is quantifiably measured. This technique actually requires that the transfer experiment be run, and is therefore not helpful in choosing tasks for transfer, but could have other conceivable uses. The exact manner in which the advantage should be quantified is one of the major challenges with d_T. We used several techniques, including the average reward received within some window of time, and the time to "convergence." Primarily we used this task similarity measure to evaluate the ability of the other task similarity measures to provide a useful approximation to d_T.

Policy overlap (d_P) finds the number of states with identical policy (maximum utility). This is perhaps the most obvious approximation to d_T, however, this can be problematical in states with two or more actions with nearly equal utility. If the action with the maximum utility differs, but the difference in utility is only slightly different, should there really be 0 policy overlap? Furthermore, differences in policy in one state may be more important than differences in another. None of these features are captured by a simple policy overlap task similarity measure. Sebastian Thrun used d_P together with a description length parameter to learn sub-skills in a suite of tasks [8].

If Q-learning is used, another simple task similarity measure can be constructed from the mean squared error between the Q-Values $Q(s, a)$ (the expected discounted future reward for taking action a in state s) of the source and target tasks. We call this task distance measure d_Q.

If the expected immediate reward for taking action a in state s (what we call the R-Values or $R(s, a)$) are stored, we can construct another task distance measure, d_R from the mean squared error between these values in the source and target tasks.

Which task similarity measure (or combination of task similarity measures) to use will depend upon the types of tasks in the library and the transfer techniques that the agent uses.

V. Task Clustering

Once a task similarity measure has been established, it is possible to use that measure to cluster the tasks in a library. There could be many advantages to clustering tasks in a library. As the agent explores a portion of the space of a target task, and finds that this portion is similar to the same portion in a group of clustered tasks, then it may be reasonable to assume that the new target task might also share similarities with this cluster of tasks in other parts of its state space. Thus clustering could simplify the process of picking a set of tasks from which to transfer. Task Clustering could even lead to the automation of this process.

Sebastian Thrun has already shown how task clustering might work with the generation of a task library system in the domain of classification tasks [9] [10]. In reinforcement learning the situation is more complex.

Task clustering may be useful in an eventual multiple task transfer mechanism because it may be easier to find a cluster of similar tasks than to determine the most similar task in a library. Furthermore, transferring features from a cluster of similar tasks may be more effective than transferring from the most similar task in the library, because the cluster of tasks may be more likely to capture invariants that all such tasks share rather than details specific to a given task [2].

We employed a simple merge clustering algorithm that first placed each task into its own cluster and then found the two tasks with the minimum distance between them, and merged them into a cluster, and then repeated. Different cluster trees were generated with different task similarity measures. This allowed the agent to capture a different set of features that the tasks have in common. By analyzing the different trees generated by different task similarity measures it is possible to compare the different properties of each task similarity measure.

VI. Methodology

In our experiments we used a complex grid world of size 50 by 50, where an agent can face one of eight directions and move either forward or backward, and turn an eighth to the left or right (for more information on this world see [10][7]). To test our task similarity measures we performed three classes of experiments, the moving goal experiments, the expanding obstacle experiments, and the clustering experiments. In all cases, the transfer mechanism employed was direct transfer (for more information on transfer mechanisms see [7]).

1) In the moving goal case we generated 74 different tasks by placing one goal in each task in varying positions chosen to create an approximately uniform distribution of tasks across the grid world maze. We picked one task, and used each of the other tasks as source tasks to speed the learning of this one target task generating d_T. We then analyzed how this advantage was approximated by the other three distance measures d_Q, d_R, and d_P. We computed the distance from each task to the target task both with the target task thoroughly learned (ran until convergence) and partly learned (prematurely stopped at 300,000 steps). This allowed us to determine how quickly the measures were able to generate their approximations.
2) For the expanding obstacle set of experiments we placed a single obstacle in the center of the complex grid world, and allowed this obstacle to vary in size from task to task. As before we tested the distance measures both thoroughly and partially learned.
3) For clustering we placed several goals in the general area of the bottom left, and several goals in the general area of the upper right. In half of the tasks we removed all the obstacles from the complex grid world, generating a large open room. In the other half we placed a single obstacle of uniform size and shape in the center of the world. This allowed us to compare each distance measure's sensitivity to general policy trends (bottom left to upper right, or upper right to bottom left) with each distance measure's sensitivity to the existence of the obstacle.

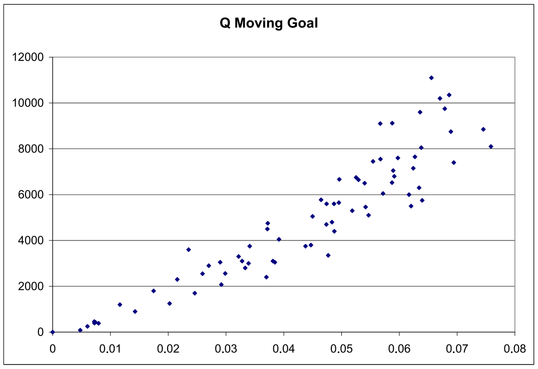

Fig. 1. d_Q on the x axis vs d_T on the y axis in terms time to convergence in 1000 step units. Smaller distance measures indicate more similarity between tasks, and therefore take less time to adapt using direct transfer. The moving goal case.

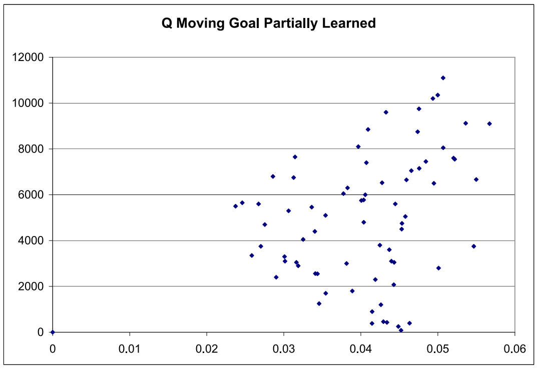

Fig. 2. d_Q on the x axis vs d_T on the y axis, after 300,000 steps. When the target task is partially learned d_Q provides a poor approximation to transfer time in the moving goal case.

VII. Evaluation

In this section we empirically evaluate the task similarity measures proposed in section IV. We evaluate each task similarity measure relative to d_T, (the actual advantage gained by using one task to improve learning of another) in the moving goal, the expanding obstacle, and task clustering cases.

A. Moving Goal Experiments

In the case of the moving goal, we used a task with the goal in the upper right corner, and found the distance between that task and the other tasks in the library with goals scattered throughout the space.

The $d_Q(i,k)$ distance measure provides a good approximation to the speedup expected when using task i to speed learning of task k, when using direct transfer as a transfer mechanism, as shown in Figure 1. Although the data is hetroscedastic, the number of iterations to learn a task rises approximately linearly with d_Q. In this experiment the source and target tasks were learned to convergence, and this represents the best possible results for d_Q. The problem with this task similarity measure is that it requires the Q-values to be known (or at least well approximated) in both the source and the target task. If the Q-values are not fully learned, d_Q does not work well, as shown in Figure 2. Since the point

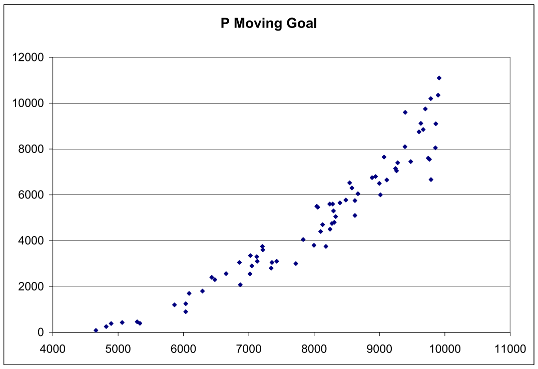

Fig. 3. d_P on the x axis vs d_T on the y axis in units of a thousand steps, the moving goal case.

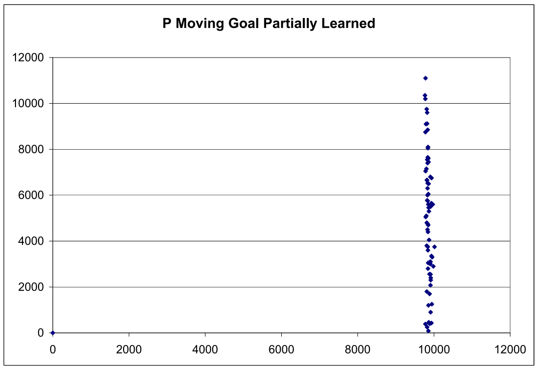

Fig. 4. d_P on the x axis vs d_T on the y axis in units of a thousand steps. When the target task is partially learned the distance measure provides a poor approximation to transfer time in the moving goal case.

of transfer is to aid in the learning of the Q-values in the target task, requiring fully learned Q-values in the target task is unrealistic.

The moving goal results for d_P are very similar to d_Q (see Figure 3). This measure also requires that the tasks are thoroughly learned before providing an accurate estimate of task similarity in the moving goal case (see Figure 4).

d_R converges long before d_Q or d_P. In general, rewards can be learned long before the correct Q-values (and the correct policies) can be propagated back through the state space. The problem with d_R in this case, is that if there are two tasks in the library, one with a reward moved a small amount from the target location and another with the reward moved a large distance from the target location, both will appear to be equally similar to the target task. In our moving goal test all of the tasks in the library are the same except for the location of the goal, the very attribute d_R can not differentiate (see Figure 5). Since d_R does not perform in this case even when thoroughly learned, it is no surprise that it also failed when partially learned (graph not shown for space).

B. Expanding Obstacle Experiments

In the expanding obstacle case, d_Q, d_P, and d_R were good approximations to d_T when the target task was thoroughly learned (see Figures 6, 8, and 10). When the target task was

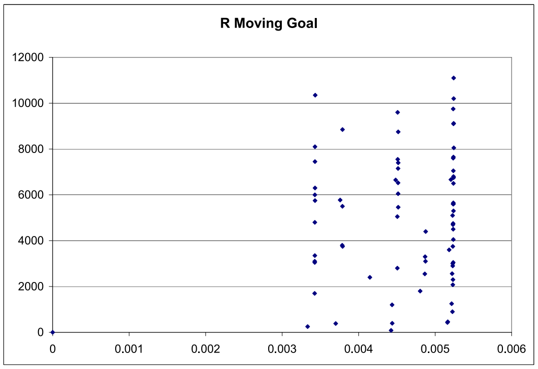

Fig. 5. d_R on the x axis vs d_T on the y axis in units of a thousand steps, the moving goal case.

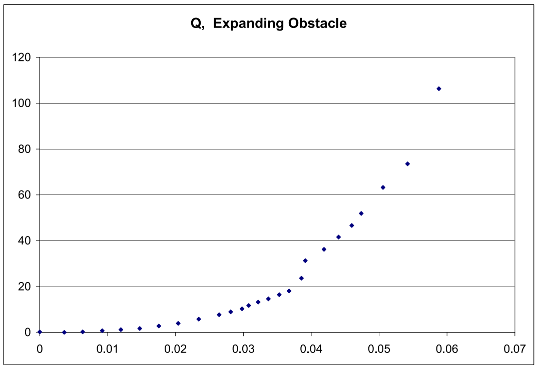

Fig. 6. d_Q on the x axis vs d_T on the y axis, computed with average reward, expanding obstacle case.

not thoroughly learned d_Q and d_R were good approximations to d_T, while d_P was not (see Figures 7, 9, and 11).

Note that d_Q was a good approximation to d_T before the Q-values in the target task were thoroughly learned in the expanding obstacle case, but failed to do so in the moving goal case. The difference between the moving goal result and expanding obstacle result can be easily explained by the following example. In the moving goal case, it often happens that there are some source tasks with a goal that is a small distance from the goal's location in the target task, and other source tasks with the goal placed a large distance from the goal's location in the target task. If the agent has only learned the Q-values close to the goals, then the distances computed will be the same regardless of the distance the goal was moved until the Q-values back up a sufficient distance. However, in the expanding obstacle case, the agent can quickly learn the Q-values going into the obstacle. Since the other Q-values will be nearly the same during the initial stages of learning, the Q-values near the obstacle dominate in the distance computation and will provide an excellent approximation to the difference in the final Q-values that would eventually be learned by the agent. Thus, in this case, the initial approximation provided by d_Q is approximately correct, while it is not correct in the moving goal case.

d_P's behavior in the expanding obstacle case is very similar to its behavior in the moving goal case when the tasks were

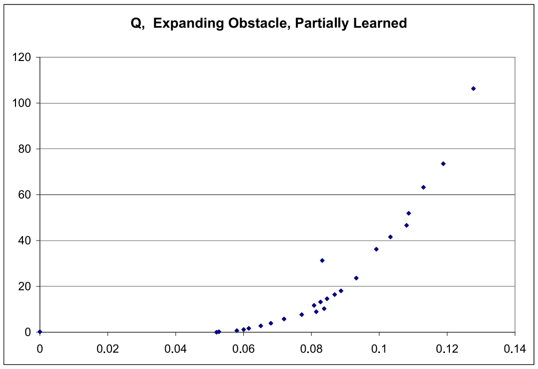

Fig. 7. d_Q on the x axis vs d_T on the y axis. When the target task is partially learned the distance measure provides a good approximation to transfer time in the expanding obstacle case.

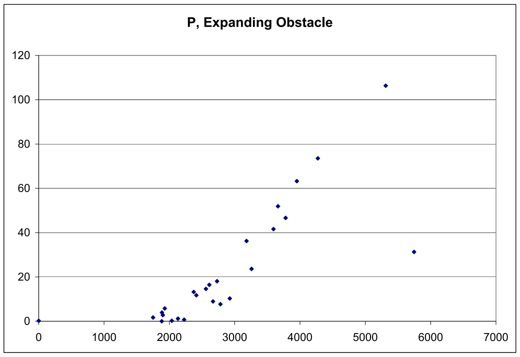

Fig. 8. d_P on the x axis vs d_T on the y axis, expanding obstacle case.

thoroughly learned, except that in our complex grid world maze there are often several optimal policies that can lead to the goal. This caused the task similarity measure to be more noisy than in the moving goal case.

Unlike d_Q, d_P was unable to provide a good approximation before the policy was thoroughly learned even in the expanding obstacle case. This makes sense because d_P measures the overlap in the optimal policy. Although the agent quickly ruled out actions that took it into the obstacle, it had not yet learned the optimal policy, and this technique doesn't take into account what it had learned about what not to do.

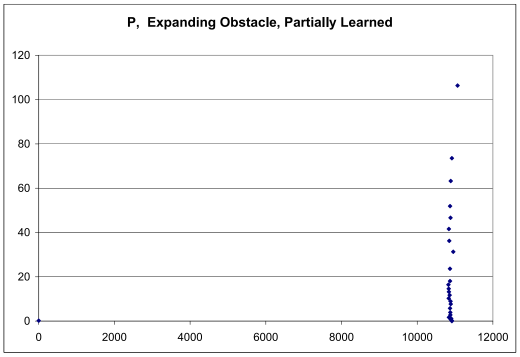

Fig. 9. d_P on the x axis vs d_T on the y axis. When the target task is partially learned $d_P(i,k)$ cannot provide a good approximation to transfer time in the expanding obstacle case, unlike $d_Q(i,k)$.

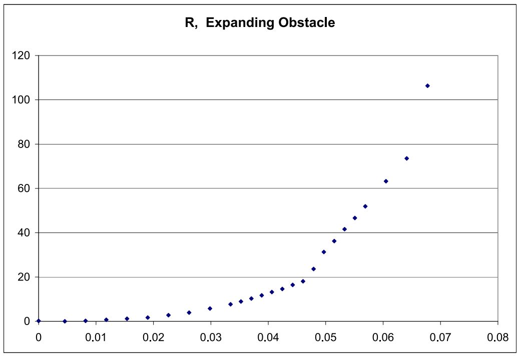

Fig. 10. d_R on the x axis vs d_T on the y axis, expanding obstacle case.

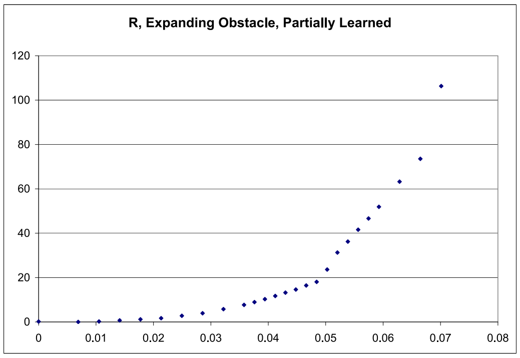

Fig. 11. d_R on the x axis vs d_T on the y axis. When the target task is partially learned $d_R(i,k)$ can provide a good approximation to transfer time in the expanding obstacle case.

In the expanding obstacle case, d_R was a good approximation to d_T, which it was not in the moving goal case. This is because in the expanding obstacle case the difference in the number of negative rewards going into the obstacle provided an excellent approximation to the transfer value of a given task. Thus this measure cannot capture the distance that a goal is moved, but can easily capture the fact that new goals or obstacles have been added to the problem.

These results are summarized in Table I.

C. Clustering

The cluster tree based upon d_Q correctly separated out tasks that had the goal near the upper right from those tasks that had the goal near the bottom left. These categories of tasks were then further broken down into tasks that had an obstacle from those tasks that had no obstacle (see figure 12). This shows that d_Q is more sensitive to the general policy of the task than it is to the presence of obstacles, yet it was able to detect the presence of the obstacles.

Like d_Q, d_P is more sensitive to the general policy than it is to the presence or absence of obstacles in the task (see

TABLE I

SUMMARY OF ENVIRONMENTS AND SUCCESSFUL TASK SIMILARITY MEASURES

	Fully learned			Partially learned		
	d_Q	d_P	d_R	d_Q	d_P	d_R
Moving goal	✓	✓				
Expanding obstacle	✓	✓	✓	✓		✓

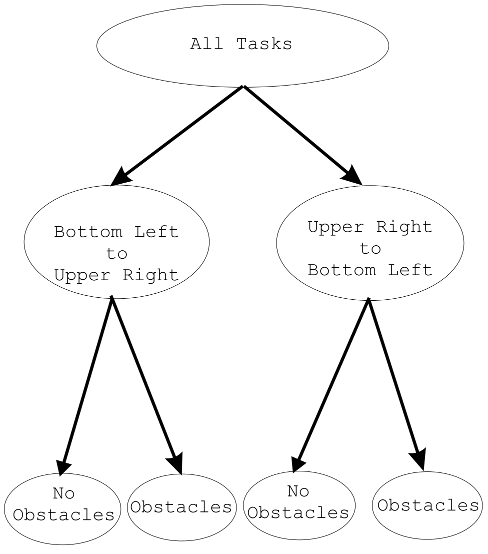

Fig. 12. Cluster tree 1 and 2, The same cluster tree was created by both d_Q and d_P.

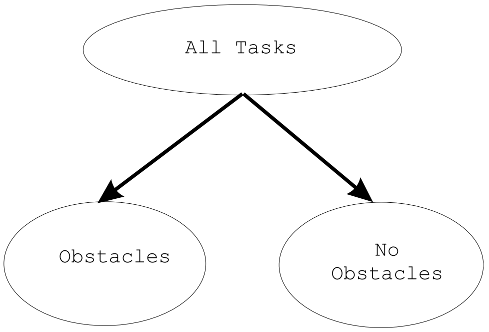

Fig. 13. Cluster tree 3, based on d_R.

Figure 12).

When using d_R the tasks were sensitive to the presence of the obstacle in the center, but were unable to capture the general policy trends (see figure 13).

VIII. Conclusion

No best measure for similarity between any two tasks in a MDP exists apart from the transfer mechanism employed. We have proposed that task similarity could be defined as the average amount of speedup expected when using one source task to speed the learning of another target task under a given transfer technique. We proposed three properties that a task similarity measure should have.

Creating approximations to such a task similarity measure that can be evaluated before the actual target task is learned can help an agent use the information in its task library to adapt to new situations more quickly. Such a measure is useful because an agent can use this measure to determine which tasks to use in transfer given a particular transfer technique. We have shown that different task similarity measures capture different types of similarity, thus an agent should have more than one transfer mechanism, which it could use to determine which transfer technique to use given a set of prior tasks.

Different distance measures often capture different types of differences between tasks. Both d_Q and d_P best capture trends in the overall policy but are only accurate after the Q-values or the policy have been learned. However, approximations to these values can be computed before the Q-values or policy have been thoroughly learned. The accuracy of the approximation will depend upon the type of task being learned, and the task similarity measure employed. In the moving goal case the task must be more thoroughly learned than in the expanding obstacle case for d_Q to provide an accurate approximation. In both cases d_P requires the task to be more thoroughly learned than does d_Q. d_R can be computed before the policy has been learned, but is less sensitive to overall policy trends. In the moving goal case d_R is not just less sensitive to over all policy trends, it is incapable of determining the difference between moving a goal a short distance vs. moving the goal a long distance.

IX. Future Research

A heuristic with the ability to return a useful approximation of task similarity before the tasks are learned, like d_R, but which captures differences in policies like d_Q or d_P should be developed.

An analogy based task similarity measure would be extremely useful, however, it is not clear how to build such a measure.

Eventually the system should compute multiple task similarity measures for the tasks in the library and then automatically select a set of tasks and transfer techniques suited to the target task.

References

[1] T. Peterson, N. Owens, and J. L. Carroll, “Automated shaping as applied to robot navigation,” in *IEEE International Conference on Robotics and Automation*, Seoul, Korea, 2001. [Online]. Available: http://james.jlcarroll.net/publications/shaping.pdf

[2] J. L. Carroll, T. Peterson, and K. Seppi, “Reinforcement learning task clustering (rltc),” in *ICMLA*, Los Angeles, CA, 2003. [Online]. Available: http://james.jlcarroll.net/publications/rltc.pdf

[3] J. L. Carroll, T. Peterson, and N. Owens, “Memory-guided exploration in reinforcement learning,” in *IJCNN*, Washington, D.C., 2001. [Online]. Available: http://james.jlcarroll.net/publications/mge9.pdf

[4] M. Bowling and M. Veloso, “Reusing learned policies between similar problems,” in *AI*AI-98 Workshop on New Trends in Robotics*, Padua, Italy, October 1998.

[5] A. Robins, “Transfer in cognition,” in *Learning to Learn*, L. P. S. Thrun, Ed. San Francisco, CA: Morgan Kaufmann, 1998, ch. 3, pp. 45–67.

[6] J. Randlov and P. Alstrom, “Learning to drive a bicycle using reinforcement learning and shaping,” in *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kaufmann, CA, 1999.

[7] J. L. Carroll, “Task localization, clustering, and transfer; towards a reinforcement learning task library system,” Master’s thesis, Brigham Young University, Provo, UT, 2005.

[8] S. Thrun and A. Schwartz, “Finding structure in reinforcement learning,” in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds., 1995, pp. 385–392.

[9] S. Thrun and J. O’Sullivan, “Clustering learning tasks and the selective cross-task transfer of knowledge,” in *Technical Report CMU-CS-95-209*. Pittsburg, PA: Carnegie Mellon University, 1995. [Online]. Available: citeseer.nj.nec.com/thrun95clustering.html

[10] ——, “Discovering structure in multiple learning tasks: The TC algorithm,” in *International Conference on Machine Learning*, Bari, Italy, 1996, pp. 489–497. [Online]. Available: citeseer.nj.nec.com/thrun96discovering.html