Knowledge Granularity Spectrum, Action Pyramid, and the Scaling Problem

Yiming Ye
IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA [a]

John K. Tsotsos
Department of Computer Science, University of Toronto, Toronto, Canada

In this paper we introduce the concept of knowledge granularity and study the relationship between different knowledge representation schemes and the scaling problem. By scale to a task, we mean that an agent's planning system and knowledge representation scheme are able to generate the range of behaviors required by the task in a timely fashion. Action selection is critical to an agent performing a task in a dynamic, unpredictable environment. Knowledge representation is central to the agent's action selection process. It is important to study how an agent should adapt its methods of representation such that its performance can scale to different task requirements. Here we address the following issue: to scale to a given task, when should an agent represent its knowledge using a single granularity and when should an agent represent its knowledge using a set of hierarchical granularities?

1 Introduction

This paper studies the scaling problem with respect to an agent - a computational system that inhabits dynamic, unpredictable environments. An agent has sensors to gather data about the environment and can interpret this data to reflect events in the environment. Furthermore, it can execute motor commands that produce effects in the environment. Usually, it has more or less knowledge about itself and the world. This knowledge can be used to guide its action selection process when exhibiting goal-directed behaviors. It is important for an agent to choose a reasonable representation scheme in order to scale to the task at hand. There are two extremes regarding granularity of knowledge representation. At one end of the spectrum is the pure reactive scheme [2] which requires little or even no knowledge representation. At the other end of the spectrum is the pure planning scheme which requires the agent to maintain and use as much detailed knowledge as possible. Experience suggests that neither purely reactive nor purely planning systems are capable of producing the *range of behaviors* required by intelligent agents in a dynamic, unpredictable environment. For example, Tyrrell [13] has noted the difficulty of applying ,

[a] in Intelligent Agent Technology -Systems, Methodologies, and Tools, pp 152-161, Editors Jiming Liu and Ning Zhong, World Scientific Publishing Co. Pte. Ltd. 1999

without modification, the model of Brooks [2] to the problem of modeling action selection in animates whose behavior is supposed to mirror that of real animals. On the other hand, although it is theoretically possible to compute the optimal action selection policy for an agent that has a fixed set of goals and that lives in a deterministic or probabilistic environment [13], it is impossible to do so in most practical situations for the following reasons: (A) resource limitations (time limit, computation complexity [15], memory limit); (B) incomplete and incorrect information (knowledge difference [16], sensor noise, etc); (C) dynamic, non-deterministic environment. Thus, many researchers argue to use hybrid architectures [14 5 10 7], a combination of classical and alternative approaches, to build agent systems. One example is the layered architecture [5 10]. In such an architecture, an agent's control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction. Thus, the very lowest layer might map raw sensor data directly onto effector outputs, while the uppermost layer deals with long-term goals. Or, the upper abstract space might be used to solve a problem and then the solution might be refined at successive levels of detail by inserting operators to achieve the conditions that were ignored in the more abstract spaces [8 9].

In this paper, we consider knowledge abstraction over a spectrum based on the granularity of knowledge representation. Our approach is different from previous approaches [5 10] in that there is no logical relationship between elements of any two adjacent layers. We study the scaling problem related to different representation schemes, be it a single granularity scheme or a hybrid granularity scheme. Much of the previous work on scaling emphasizes the absolute complexities (efficiency) of planning systems. We, however, offers an alternative point of view: we believe that the problem of scaling is a relative term and is closely related to the task requirements of an agent in uncertain, dynamic or real-time environments. We will say that an agent *scales* to a given task, if the agent's planning system and knowledge representation scheme are able to generate the range of behaviors required by the task. Many factors, such as the planning engine, the way knowledge is represented, and the dynamic environment can influence whether an agent scales to a given task. In this paper, we concentrate on the influences of knowledge granularity. It is obvious that knowledge granularity can influence the efficiency of a given inference engine, since granularity influences the amount of data to be processed by the engine. It has been suggested that one may increase the computational efficiency by limiting the form of the statements in the knowledge base [11 4]. Here, we study the relationship between different representation schemes and the performance of an agent's planning system. The goal is to find the proper scheme for representing an agent's knowledge such that the representation

allows the agent to scale to a given task. In [17], we studied the problem of how to define the granularity of an agent's representation of a certain kind of knowledge. In [18], we studied how this granularity influences the agent's action selection performance. In this paper, we relate the scaling problem of an agent with its knowledge granularity and study how the hierarchical granularity representation influences the agent's action selection performance. The results of this paper can help an agent in finding a reasonable granularity or scheme of representation such that its behavior can scale to a given task.

2 Knowledge Granularity

We define knowledge granularity with respect to a certain kind of knowledge as the total memory used by the agent to represent the knowledge divided by the memory used by the agent to represent a basic element of the corresponding knowledge. For example, suppose the task of an agent is to search for an object within a two dimensional square of size 100 (unit) $\times$ 100 unit. The agent encodes its knowledge about the possible position of the target as probability distributions within the square and selects actions based on this distribution. If the agent tessellates the square into 100×100 small squares with size 1 (unit) $\times$ 1 (unit) and encodes probabilities, then the corresponding knowledge granularity g is $\frac{100 \times 100 \times m[p(c)]}{m[p(c)]} = 10,000$. Where $p(c)$ is the probability associated with any small square c and $m[p(c)]$ is the memory used by the agent to represent $p(c)$. It is obvious that knowledge granularity influences the performance of the agent. For a fixed action selection algorithm, a higher granularity usually results in a better selected action because the encoded knowledge are usually more accurate. However, the action selection time is usually longer because the algorithm has more items to manipulate.

It is very interesting to study how the performance of an agent is influenced by granularity, and how an agent should choose a reasonable granularity from the spectrum of knowledge abstraction for different time constraint T, the total time available for the agent to perform its task. Suppose for a granularity g, the *average* time needed to select an action is $t_s(g)$, the *average* contributions of a selected action to the task is $Q(g)$. *Assuming* that the total contributions U made by an agent within the time constraint T can be represented by the sum of the average contributions of all the actions that is executed within T. Then, U can be represented as follows:

$$U(g) = \lfloor \frac{T}{t_e + t_s(g)} \rfloor Q(g).$$

Where t_e is the time needed by the agent to execute an action, and $\lfloor \frac{T}{t_e + t_s(g)} \rfloor$

gives the total number of actions that can be selected and executed by the agent within T. Obviously, the best granularity g is the one that maximizes $U(g)$. Different agents use different kinds of knowledge and different kinds of action selection procedures. Because of the complexity and diversity of the world of agents, it is impossible to provide a general solution for g. What we can do is to group agents into different categories and study the behavior with respect to each category. For example, let $f_1(g) = 5$; $f_2(g) = \ln(g)$; $f_3(g) = g + 1$; $f_4(g) = g^3 + 1$; $f_5(g) = \exp(g) + 1$. These functions represent a small subset of different degrees of the influence of g on the entities to be discussed. Please note that function $f_1(g)$ means that the entity to be discussed is a constant, and thus is not influenced by g. In real situations, the relationship between g and the entities might be much more complex. In [18], we studied how $U(g)$ is influenced by g by setting $t_s(g)$ and $Q(g)$ equal to different functions as listed above. The message derived from our study shows that knowledge granularity has a big impact on the performance of an agent. Thus, an appropriate knowledge granularity should be selected by an agent in order to guarantee a satisfactory result. We also suggested ways of selecting a reasonable granularity when a single granularity is used.

3 Knowledge Granularity Spectrum and Action Pyramid

Usually, with respect to the knowledge granularity spectrum, the higher the value of the knowledge granularity, the better the quality of the selected actions. However, it is not always beneficial to use high granularity, because the cost usually increases as well. In order to benefit from both the short action selection time of low granularity and the high quality of the selected actions of high granularity, a hierarchy of granularity layers can be used to select actions. For example, we can choose several granularities for the purpose of action selection, as follows. First, the coarsest granularity is used to select a set of actions from the pool of all the actions. Then the second coarsest granularity is used to select a even smaller set of actions from the set of actions chosen before. This procedure continues until the actual action to be applied is selected according to the finest granularity. The sets of actions selected by different granularities form an action pyramid. In this section, we compare the performances of a single layer granularity scheme and the multi-layer granularity scheme. We restrict our discussion to two layers because of limited space. Similar results can be obtained for more than two layers.

3.1 *Adding a coarse new layer for filtering out non-interesting actions*

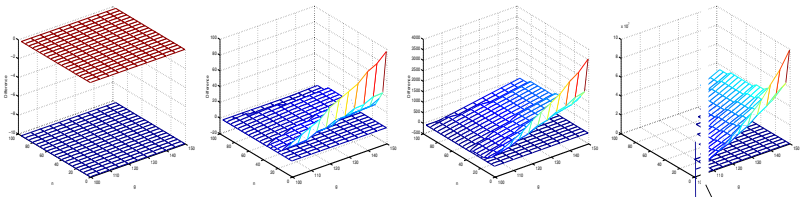
Suppose after some analysis we find that granularity g_o is a favorable choice for a single granularity scheme. We might increase the performance by adding a coarser layer. The idea is to use the coarser granularity to select a small set of actions that are suitable, and then use g_o to select an action to execute from this small set of actions. Suppose originally there are totally N actions to be selected from. Now consider adding another layer of granularity g_c. Suppose that for g_c, we need to collect the first n_{g_c} actions in order to guarantee that enough good actions are within this set of actions. In other words, the quality of the actions selected by g_o from the limited n_{g_c} actions is almost the same as the quality of the actions selected by g_o from the N actions.

Suppose that for this planning system the action selection time is governed by $t(g, n)$, where g is the granularity and n is the number of actions to be considered. Now we compare the time needed to select one action for the two strategies. The time, T_o^s, needed to select an action from the single layer is: $T_o^s = t(g_o, N)$. The time, T_n^s, needed to select an action for the new strategy is: $T_n^s = t(g_c, N) + t(g_o, n_{g_c})$. Thus, the difference in selecting an action by the two strategy is given by:

$$T^s = T_n^s - T_o^s = t(g_c, N) + t(g_o, n_{g_c}) - t(g_o, N)$$

Figure 1 shows the results of experiments that are performed to test the performance difference under different strategies. It draws surfaces of $T^s = T_n^s - T_o^s$ as a function of n_{g_c} and g_c under various situations. In the test, we set $N = 100$ and $g_o = 100$. The range for n_{g_c} is $1 \leq n_{g_c} \leq 100$, the range for g_c is $15 \leq g_c \leq 100$. In order to make the comparison easier, we also draws the surfaces of $T^s = 0$. For any pair n_{g_c} and g_c, if the corresponding point on the surface is above $T^s = 0$, then the two layer strategy is not a favorable choice, because it needs more time to select an action; otherwise, the two layer strategy should be used because it spends less time in selecting an action. In our experiments, we assume that $t(g, n) = t_1(g)t_2(n)$. Where the function t_1 gives the sensitivity measurement of the granularity, g, with respect to the action selection time, and t_2 gives the sensitivity measurement of the number of actions to be considered, n, with respect to the action selection time. The index (i, j) in Figure 1 means that $t_1(x) = f_i(x)$, $t_2(y) = f_j(y)$, where function f_i is defined in the previous section. For example, Figure 1(2,3) means that $t_1(x) = f_2(x)$, $t_2(y) = f_3(y)$. Figure 1(1,1) shows that when $t(g, n)$ is not influenced by g and n, the two layer strategy is always worse than the single layer strategy by a constant. This constant is the time used to pre-select the set of actions by the coarse layer. Figure 1(1,1)(1,2)(1,3) show the situation when

the action selection time is not influence by granularity. In this case, adding a new coarse layer does not save time. Because the coarse layer itself will spend the same time as the old granularity g_o, and extra time must be spent by g_o to select an action from the pre-selected action pools by g_c. Figures 1(2,1)(2,2)(2,3)(2,4)(2,5) shows that the more sensitive the action selection time is influenced by the number of actions in the action pool, the better the two layer strategy. This is illustrated by the increase of the area of those g_c and n_{g_c} that are below the plane $T^s = 0$. The reason is that a decrease in granularity for a more sensitive situation tends to have a bigger saving in action selection time. The same analysis can be applied to Figures 1(3,2)(3,3)(3,4) and Figures 1(4,1)(4,2)(4,3)(4,4)(4,5). From Figure 1, we can also notice that for a fixed granularity g_c, the smaller the value of n_{g_c}, the better the two layer strategy. The reason is that a smaller n_{g_c} tends to save time for g_o. We can also notice that for a fixed n_{g_c}, the smaller the value of g_c, the better the two layer strategy.



From above experiment, we know that in some situations adding a coarse layer can increase the performance of an agent. Thus, when a single granularity does not allow the agent to scale to the task at hand, we can consider adding a coarse layer to increase the chances of scaling. To do this, we can first draw the performance figure as above, and then select the granularity that corresponds to the lowest point on the surface as the granularity for the coarse layer.

3.2 Adding a finer layer to obtain better quality actions

Another way to use hierarchical representation to increase performance and chances of scaling is to add a finer layer. The idea is to use the current granularity g_o to pre-select a small set of candidate actions, and then use a finer granularity g_f to choose a better quality action to execute.

The total utility for the single layer strategy within time constraint T is:

$$U(g_o) = \lfloor \frac{T}{t_e + t_s(g_o, N)} \rfloor Q(g_o)$$

For the two layer strategy, Suppose n_{g_o} is the number of actions that must be selected by g_o in order to guarantee that the actions selected by g_f will reach a desired quality $Q(g_f)$. The time to select an action for the two layer strategy is: $t_s = t(g_o, N) + t(g_f, n_{g_o})$. The utility of the new strategy is:

$$U(g_f) = \lfloor \frac{T}{t_e + t_s(g_o, N) + t_s(g_f, n_{g_o})} \rfloor Q(g_f)$$

Experiments have been performed to show the performance difference of the new strategy and the old strategy, $U_{diff} = U(g_f) - U(g_o)$. We assume: $T = 100$, $g_o = 100$, $N = 100$. We also assume that the action execution time is $t_e = 6$. In general t_e has a big influence on the analyzing result. Here we take $t_e = 6$ as an example to study the influence of other factors on the agent performance.

In Figure 2, we draw the surfaces of U_{diff} as a function of g_f and n_{g_o}. The range for g_f and n_{g_o} are $[100, 150]$ and $[1, 100]$ respectively. We assume $t_s(g, n) = t_1(g)t_2(n)$. $Q(g)$ is another function, which gives the quality of the action selected with granularity g. In Figure 2, the index (i, j, k) means that the figure is drawn by setting $t_s(g, n) = f_i(g)f_j(n/70)$ and $Q(g) = f_k(g)$. Figure 2(1,1,1) shows the situation when granularity and the number of actions to be selected has no influence on the action selection time, and granularity has no influence on the quality of the selected actions. In this situation, the two layer strategy is always worse than the one layer strategy, because it is a waste of effort to pre-select a set of actions for the finer layer. Figures 2 (2,2,2) (2,2,3)

(2,2,5), Figures 2 (2,3,2) (2,3,3) (2,3,5), Figures 2 (3,3,2) (3,3,3) (3,3,5), and Figures 2 (3,5,2) (3,5,3) show that the more sensitive the quality of selected actions with respect to granularity, the more benefit the two layer strategy. The influence of the sensitivity of n_{g_o} with respect to action selection time on the performance can be complex, as shown in Figures 2 (2,2,2) (2,3,2) (3,3,2) (3,5,2) (2,2,3) (2,3,3). In general, many factors can influence the performance of the two strategies and a graph need to be drawn in order to determine which strategy is better.

4 Conclusion

In this paper, we introduce the concept of knowledge granularity and study the relationship between different knowledge representation schemes and the scaling problem. We promote the viewpoint that the problem of scaling is closely related to an agent's task requirement. By scale to a task, we mean that an agent's planning system and knowledge representation scheme are able to generate the range of behaviors required by the task in a timely fashion. Here, we study the influence of knowledge granularity and related representation schemes on an agent's scaling problem. We conduct experiments to compare the performance between a single layer granularity scheme and multiple layer granularity scheme. The reason to use several granularities from the granularity spectrum and to choose actions from the corresponding action pyramid is that the agent might benefit from both the short action selection time of low granularities and the high quality of the selected actions of high granularities. Experimental results show that a hierarchical representation scheme can produce a better performance in some situations, especially when the quality of the selected actions are greatly influenced by the granularity or when the action selection time is greatly influenced by the granularity.

References

1. R. Bajcsy. Active perception vs. passive perception. In *Third IEEE Workshop on Vision*, pages 55–59, Bellaire, 1985.
2. R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, (47):139–160, 1991.
3. Connel. *An Artificial Creature*. PhD thesis, AI Lab, MIT, 1989.
4. J. Doyle and R. Patil. Two theses of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence*, 48:261–298, 1991.
5. I. Ferguson. *Touring Machine: An Architecture for Dynamic, Rational, Mobile Agents*. PhD thesis, University of Cambridge, UK, 1992.

6. T. D. Garvey. Perceptual strategies for purposive vision. Technical Report Technical Note 117, SRI International, 1976.
7. M. Georgeff and A. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 677–682, Seattle, WA, 1987.
8. F. Giunchiglia, A. Villafiorita, and T. Walsh. Theories of abstraction. *AI Communications*, 10:167–176, December 1997.
9. C. Knoblock. Automatically generating abstractions for planning. *Artificial Intelligence*, 68:243–302, 1994.
10. J. Muller and M. Pischel. Modelling interacting agents in dynamic environments. In *Proceedings of the Eleventh European Conference on Artificial Intelligence*, pages 709–713, Amsterdam, The Netherland, 1994.
11. B. Selman and H. Kautz. Knowledge compilation and theory approximation. *Journal of the ACM*, 43(2):193–224, March 1996.
12. S. Sen, S. Roychowdhury, and N. Arora. Effects of local information on group behavior. In *Proceedings of Second International Conference on Multi-Agent Systems*, pages 315–321, Kyoto, Japan, 1996.
13. T. Tyrrell. *Computational Mechanisms for Action Selection*. PhD thesis, Center for Cognitive Science, University of Edinburgh, England, 1993.
14. M. Wooldridge and N. Jenning. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.
15. Y. Ye and J. K. Tsotsos. Sensor planning in 3d object search: its formulation and complexity. In *The 4th International Symposium on Artificial Intelligence and Mathematics*, Florida, U.S.A., January 3-5 1996.
16. Y. Ye and J. K. Tsotsos. Knowledge difference and its influence on a search agent. In *First International Conference on AUTONOMOUS AGENTS*, Marina del Rey, CA, January 1997b.
17. Y. Ye and J. K. Tsotsos. Knowledge granularity and action selection. In *Eighth International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, pages 475–489, 1998.
18. Y. Ye and J. K. Tsotsos. Knowledge granularity for task oriented agents. In *Proceedings of the Third Annual Conference on Autonomous Agents*, USA, 1999.
19. Y. Ye and J. K. Tsotsos. Sensor planning for 3d object search. *Computer Vision and Image Understanding*, 73(2):145–169, February 1999.