25-7-2005

A Coloured Petri Net Based Strategy for Multi-Agent Scheduling

Q. Bai
University of Wollongong, quan@uow.edu.au

Minjie Zhang
University of Wollongong, minjie@uow.edu.au

H. Zhang
Chinese Academy of Science, China

A Coloured Petri Net Based Strategy for Multi-Agent Scheduling

Abstract

In the last decade, the focus of agent research has shifted from single agent systems to multi-agent systems (MASs). Dynamic agent coordination is one of the challenge problems of multi-agent research. One coordination problem is how to achieve agent scheduling under open dynamic environments. Petri Nets (PNs) and Coloured Petri Nets (CPNs) are system study tools that provide an appropriate mathematical formalism for the description, construction and analysis of distributed and concurrent systems. In this paper, we present a CPN based strategy to schedule and allocate new tasks to suitable agent(s) or agent combinations. In this strategy, through using CPNs to represent the dynamic statuses of agents, agent coordinators are able to check concurrent agent statuses and make correct and optimal decisions.

Disciplines

Physical Sciences and Mathematics

Publication Details

This article was originally published as: Bai, Q, Zhang, M & Zhang, H, A Coloured Petri Net Based Strategy for Multi-Agent Scheduling, Proceedings of the Rational, Robust, and Secure Negotiation Mechanisms in Multi-Agent Systems (RRS'05), 25 July 2005, 3-10. Copyright IEEE 2005.

A Coloured Petri Net Based Strategy for Multi-agent Scheduling

Quan Bai[1], Minjie Zhang[1] and Haijun Zhang[2]
[1]*School of IT and Computer Science,University of Wollongong*
{qb92, minjie}@uow.edu.au
[2]*Institute of Computing Technology, Chinese Academy of Science*
zhanghj@ics.ict.ac.cn

Abstract

In the last decade, the focus of agent research has shifted from single agent systems to multi-agent systems (MASs). Dynamic agent coordination is one of the challenge problems of multi-agent research. One coordination problem is how to achieve agent scheduling under open dynamic environments. Petri Nets (PNs) and Coloured Petri Nets (CPNs) are system study tools that provide an appropriate mathematical formalism for the description, construction and analysis of distributed and concurrent systems. In this paper, we present a CPN based strategy to schedule and allocate new tasks to suitable agent(s) or agent combinations. In this strategy, through using CPNs to represent the dynamic statuses of agents, agent coordinators are able to check concurrent agent statuses and make correct and optimal decisions.

1. Introduction

In the last decade, the focus of agent research has shifted from single agent systems to multi-agent systems (MASs). This is because capability and resource limitations of single agents make them unable to solve complex problems individually [4]. Currently, certain MASs, such as information assistants [12] and supply chain systems [10], need to operate in open, vast and distributed environments. These circumstances require agents of the system to form loosely coupled cooperation relationships. On the other hand, many applications require MASs to include various agents to work together. These agents could be heterogeneous, self-interested and may possess high-level autonomy/intelligence [4, 5]. Open environments, heterogeneity and intelligence create difficult challenges for developing multi-agent coordination strategies.

One challenge of current multi-agent system (MAS) research is that in open environments, how to find out the most suitable agent(s) or agent combination to achieve new tasks according to the current statuses and desires of agents. Toward this challenge, in this paper, we present a Coloured Petri Net (CPN) based strategy to select suitable agent(s) or agent combinations to accomplish tasks under open environments.

Petri Nets (PNs) [7, 9] and Coloured Petri Nets (CPNs) [2, 3] are system study tools that provide an appropriate mathematical formalism for the description, construction and analysis of distributed and concurrent systems. They can express statuses of concurrent systems in graphical representations and well-defined semantics, and allow formal analysis of future system status transformations [3]. In the strategy proposed in this paper, we use CPNs to represent the dynamic statuses of agents and the tasks of the MAS will be allocated to agent(s) or agent combinations according to their statuses. Here, we take Production Scheduling problems [11] as a sample application domain to explain the strategy. However, the strategy can be extended in wider problems.

Proceedings of the Rational, Robust, and Secure Negotiation Mechanisms in Multi-Agent Systems (RRS'05)
0-7695-2480-X/05 $20.00 © 2005 IEEE

The remainder of the paper is arranged as follows. The description about PNs, CPNs, and PN theories that we used in this strategy are briefly introduced in Section 2. The CPN based strategy for multi-agent scheduling is presented in Section 3. Finally, conclusions and future direction of this work are presented in Section 4.

2. Petri Nets and Coloured Petri Nets

In this strategy, we use Coloured Petri Nets (CPNs), which is a kind of high level Petri Nets (PNs), to model and represent the dynamic statuses of agents. PNs and CPNs provide a framework for the construction and analysis of distributed and concurrent systems. A Petri Net (PN)/Coloured Petri Net (CPN) model of a system describes the states, which the system may be in, and the transitions between these states. In the following subsection, we will briefly describe the basic concepts of PNs and CPNs.

The basic structure of a PN can be formally defined by a 4-tuple (P, T, A, N) (see Figure 1), where P is a set of Places, such as P1, P2, P3 and P4; T is a set of Transitions, such as T1, T2 and T3; A is a set of Arcs, such as the arc from P1 to T1, T1 to P1, P2 to T1, etc.; and N is a set of Token, for example, in Figure 1, P1 and P3 have one token in the initial state. Net structure and transition firing rules are associated together to describe how system states transfer. There are a number of transition firing rules associated with different types of PNs. However, all kinds PNs share a common firing property: a transition can be fired if the token number of all input places is equal to or greater than their arcs' weights [7]. After a transition is fired, the tokens of its input places will be moved to its output places.

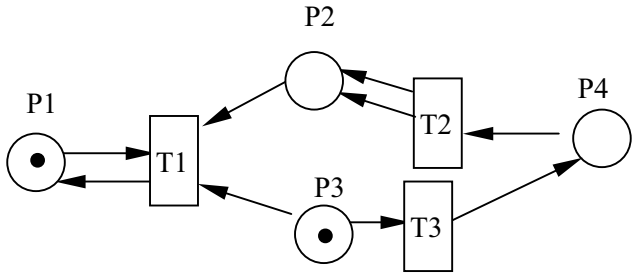

Figure 1. An example of PNs

A CPN can be defined by a 9-tuple (Σ, P, T, A, N, C, G, E, I), where Σ is a set of non-empty types, also called Colored sets; P is a set of Places; T is a set of Transitions; A is a set of Arcs; N is a node function; C is a color function; G is a guard function; E is an arc expression function; and I is an initialization function. CPNs differ from PNs because their tokens are not simply blank markers, but have data associated with them. A token's color is a schema or specification. Places of CPNs contain multi-sets of tokens. Arcs of CPNs specify the token/tokens that they can carry and can also specify some transfer conditions. Arcs exiting and entering a place can have an associated constrain function to determine which multi-set elements are to removed or hold. Transitions of CPNs are associated with some guard functions that enforce some constraints on tokens.

According to the Matrix Equation Method [7] of PN theory, the input and output functions of a PN model can be expressed as two matrixes D^- and D^+, which are called *Input Matrix* and *Output Matrix*, respectively. The subtraction result of D^+ and D^- is defined as *Composite Change Matrix* (CCM) D, where $D = D^+ - D^-$. Furthermore, in the matrix equation method, the marking of a PN model is represented as a *Marking Set* (MS) $\mu = (m_1, m_2, \ldots, m_n)$, where m_n

represents the token number of the nth place. For instance, in Figure 1, the input and output functions can be described as Equation 1, and the marking set will be μ= (1, 0, 1, 0).

$$D = D^{+} - D^{-} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & +2 & 0 & -1 \\ 0 & 0 & -1 & +1 \end{bmatrix} \quad (1)$$

With CCM and MS, we can analyse future states of a PN model by using following Equation.

$$\mu' = \mu + f(\sigma) \cdot D \quad (2)$$

In Equation 2, σ is a sequence of transition firings, $f(\sigma)$ is called firing vector, and μ' is the marking set after σ is fired. Take Figure 1 as an example, if we want to know the marking of the PN after T3 is fired, we can let $\sigma=t3$, and calculate μ' by following Equation 2, then we will have μ'= (1,0,0,1) (see Equation 3), which means a token will move from P3 to P4 after T3 is fired.

$$\begin{aligned} \mu' &= (1,0,1,0) + (0,0,1) \cdot \begin{bmatrix} 0 & -1 & -1 & 0 \\ 0 & +2 & 0 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \\ &= (1,0,1,0) + (0,0,-1,1) \\ &= (1,0,0,1) \end{aligned} \quad (3)$$

3. A CPN based strategy for multi-agent scheduling

It is becoming consensus that CPNs are one of the best ways to model concurrent systems. In the context of AI, there are a number of works of using PNs or CPNs to model agent systems and MASs. Mariusz describes a layered approach based on CPNs that can be used for modelling complex, concurrent conversations among agents in a multi-agent system [6]. Cost proposes the use of CPNs as a model underlying a language for protocol specification by taking the advantages of CPNs' great expressive power to support for concurrency [1]. Poutakidis uses Petri Nets to monitor agent conversations and to provide precise and informative error messages when agent communication protocols are not correctly followed by the agents [8]. In the strategy presented in this paper, we will use CPNs to represent statuses of agents of a MAS.

3.1. The scenario: production scheduling systems

In this paper, we choose production scheduling as the application domain of the strategy. In recent years, MASs have been successfully applied to solve many problems of manufacturing areas. A Production Scheduling Multi-agent System (PSMAS) [11] is a MAS to handle that can autonomously handle production scheduling problems. The structure of a PSMAS is shown in Figure 2.

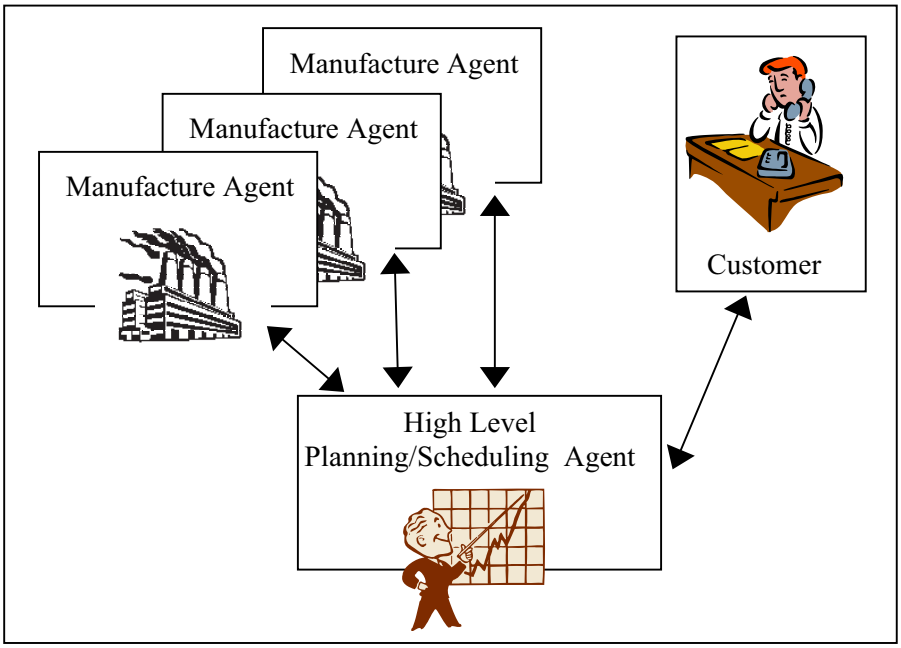

Figure 2. The structure of PSMAS

In a PSMAS, Manufacture Agents (MAs) are in charge of maintaining products manufacturing processes of subsidiary factories of a company and the Planning/Scheduling Agent (PSA), which is a higher level coordination agent, is in charge of allocating production resources and tasks to subsidiary factories, and deciding whether to accept indents from customers. To make correct decisions, the PSA must aware the statuses of MAs. Therefore, a formal representation format of MAs' statuses is necessary to enable PSA to understand. In this paper, we hire CPNs to formally represent agent's statuses.

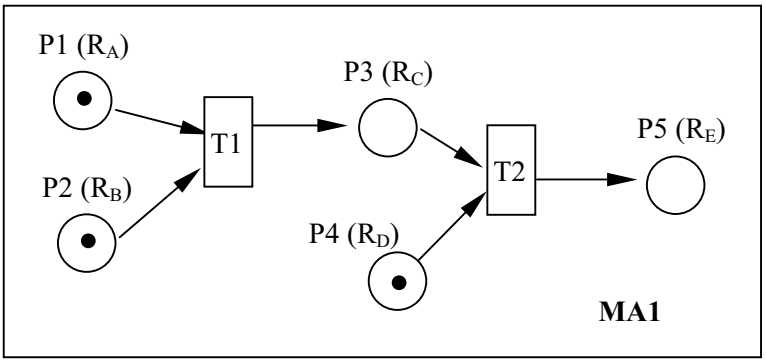

Figure 3. An example of CPN representation of MA status

3.2. Use CPNs to represent agent status

By using CPNs, a MA's resource-product relationship can be modeled as a net of components. The transitions of CPNs can be used to represent producing processes, input

places of a transition can be used to represent resources required for a producing process, output places can be used to represent output product of a producing process, and tokens in input/output places represent units of resource/product that currently in the places. For example, the Figure 3 shows the resource-product relationship of a manufacture agent MA1. In this example, input places P1 and P2 conduct resources (R_A and R_B) for process T1; the output of T1 is R_C, which is contained in P3, at the same time R_C and R_D are resources for process T2; at last, the final output product of MA1 is R_E, which is contained in P5. The status of MA1 is represented by the marking allocation of the CPN. In the current moment of Figure 1, there is one token in P1, P2 and P4, which means the factory currently has one unit R_A, R_B and R_D.

3.3. The scheduling strategy

Using CPNs to represent resource-product relationships and resource allocation statuses of MAs can enable the PSA to analyse MAs' statuses and make precise decisions on following problems:

(1) Whether the indent from a customer is acceptable;
(2) Which MA(s) can accomplish the indent;
(3) If a single MA cannot achieve an indent, which MA combination can do it?
(4) Which way is the optimal way (with lowest cost) to accomplish an indent.

Before introducing the detailed strategy, we first give following definitions.

Definition 1. Output Set (OS): The OS of a MA is the set of output product of all producing procedures of the MA. For example, in Figure 3, the OS= (O_C, O_E).

Definition 2. Total Output Set (TOS): If a PSMAS has n MAs, and the OS of each MA is OS_i ($1 \le i \le n$), the TOS of the PSMAS is the combination of OS_i, where TOS= $OS_1 \cup OS_2 \cup \ldots \cup OS_n$. $TOS = \bigcup_{1 \le i \le n} OS_i$

Definition 3. Request Product Set (RPS): If a PSMAS receive an indent from a customer, the RPS of the indent is the set of required products that included in the indent.

Definition 4. Lacking Set (LS): The LS of a MA is the set of lacking resources that embarrass some producing processes.

3.3.1. Production scope analysis: The first problem of production scheduling is to check whether the required products of the indent are with in the production scope of the company. In this strategy, if RPS$\subseteq$TOS, we can say that the indent is with in the production scope of the system. Otherwise the indent will be out of the production scope of the company and the indent cannot be accepted. For example, Figure 4 shows the current statues of manufacture agent MA1, MA2, MA3 and MA4, which are belong to production scheduling MAS PSMAS1.

According to Definition 1 and Definition 2, we can know that the TOS of PSMAS1 is (R_C, R_D, R_E, R_H, R_K, R_L). If PSMAS1 receives two indents I1 (RPS_1= (R_E, R_X)) and I2 (RPS_2= (R_E)), we can find that I1 is out of the production scope of PSMAS1 because $RPS_1 \not\subseteq$TOS.

3.3.2. Task allocation and MA status analysis: Even if RPS$\subseteq$TOS, we still cannot say that the company can accept the indent. The PSA needs to apply further analysis to find out which agent(s) is/are suitable for the task and whether the current status of the agent allows it to accomplish the task. In this strategy, this analysis is achieved through calculating the future markings of MA CPNs. For example, suppose the matrix equations of CPNs of Figure 4 are D_1, D_2, D_3 and D_4, respectively, and PSMAS1 receives an indent with RPS= (R_D, R_H), and we want to analyse whether the current status of MA2 can satisfy the indent (we do not need to analyse other MAs because only MA2 can output R_D and R_H. According to the matrix equation

method, which is introduced in Section 2, we can achieve the analysis through calculating Equation 4, and we can find that the solution is x= (1, 1, 0). Hence, the indent can be accomplished by MA2 and the required processes are T1 and T2. Therefore, the producing task can be allocated to MA2.

$$\mu' = \mu + x \cdot D_2$$

$$(0,0,1,1,0) = (1,1,0,0,0) + x \cdot \begin{bmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{bmatrix} \qquad (4)$$

$$x = (1,1,0)$$

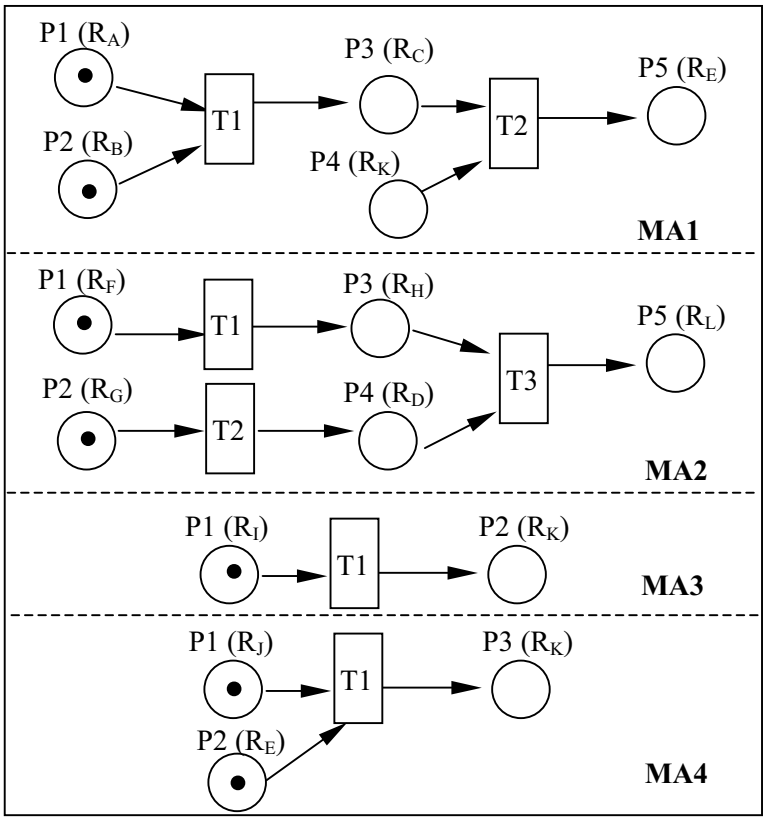

Figure 4. CPNs of MA statuses

In the example of last paragraph, the producing task can be accomplished by a MA individually. However, in some more complex cases, the producing tasks need to be achieved by more than one agent. For instance, PSMAS1 received an indent where RPS=(R_E). MA1 is the only agent that can produce R_E. However, with current marking set, we cannot find a solution for x through the same calculation as Equation 4. This indicates that MA1 is lacking some resources to operate the task, so it cannot accomplish the producing task individually. Then, we can analyze whether we can find some other MAs to supply the lacking resources and make a MA combination to achieve the task. Firstly, we have to find out the LS of the agent. In this example, MA1 cannot accept the task due to lacking of R_K, so LS= (R_K). Through checking Output Sets (OSs) of MAs we can find that R_K is an output of MA3. In

addition, MA3 can produce R_K with its current status. Therefore a combination of MA1 and MA3 is generated to execute the task together.

3.3.3. Optimal task allocation: Sometime, more than one MSs of a PSMAS can execute a task at the same time. In this case, the PSA needs to be able to allocate the task to the most suitable MA. Different Production Scheduling Multi-agent Systems (PSMASs) may have different optimal solution standards. However, production cost is one of the common standards that most PSMAS developers need to consider. With CPNs, we can easily prognosticate producing processes (transitions) and resource consumptions (places) of a production. Hence, we also can calculate the cost of a production. For instance, in PSMAS1 (see Figure 4), both MA3 and MA4 can produce R_K. If both of these two agents have enough resource to accomplish the production, which one is the optimal choice? We can find the resource consumptions of MA3 and MA4 from their input matrixes (D_3^- and D_3^-). The sequence of transitions, which represent the sequence of production processes, can be calculated by using Equation 1 and 2. Therefore, if the price of each kind of resources and the cost of each producing processes are available, the production cost can be calculated. Furthermore, comparing the production cost of MA3 and MA4, the PSA can find out the optimal solution.

4. Conclusion

In conclusion, dynamic agent coordination is one of the challenge problems of multi-agent research. One coordination problem is how to achieve agent scheduling under open dynamic environments. In this paper, we present a CPN based strategy to schedule and allocate new tasks to suitable agent(s) or agent combinations. CPNs provide an appropriate mathematical formalism for the description, construction and analysis of distributed and concurrent systems. Here, we use CPNs to represent the dynamic statuses of agents that can be checked by agent coordinators at all times. This feature can facilitate agent coordinators make correct and optimal decisions. In this paper, Production Scheduling Problems are chosen as the sample application domain to demonstrate the strategy. However, the strategy can be applied in many similar applications such as meta-computing scheduling, multi-agent resource allocation and supply chain management.

Acknowledgements

This research was supported by an International Linkage Grant from the Australian Research Council (contract number LX0346249).

References

[1] R. Cost, "Modelling Agent Conversations with Coloured Petri Nets", Working Notes of the Workshop on Specifying and Implementing Conversation Policies, Seattle, Washington, 1999, pp. 59-66.

[2] K. Jensen, "An Introduction to the Practical Use of Coloured Petri Nets", Lectures on Petri Nets II: Applications, Lecture Notes in Computer Science, Vol. 1492, Springer-Verlag, 1998, pp. 237-292.

[3] K. Jensen, Colored Petri Nets – Basic Concepts, Analysis Methods and Practical Use, Vol. 1: Basic Concepts. Springer-Verlag, Berlin, 1992.

[4] V. Lesser, "Cooperative Multiagent Systems: A Personal View of the State of the Art", IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, 1999, pp. 133-142.

[5] V. Lesser, "Reflections on the Nature of Multi-agent Coordination and Its Implications for an Agent Architecture", Autonomous Agents and Multi-agent Systems, Vol. 1, 1998, pp. 89-111.

[6] M. Nowostawski, M. Purvis and S. Cranefield, "A Layered Approach for Modelling Agent Conversations", Proceedings of the 2nd International Workshop on Infrastructure for Agents, MAS, and Scalable MAS, Montreal Canada, 2001, pp. 163-170.

[7] J. Peterson, Petri Net Theory and The Modeling of Systems, Prentice-Hall, Inc., N.J, 1981.

[8] D. Poutakidis, L. Padgham and M. Winikoff, "Debugging Multi-Agent Systems Using Design Artefacts: The Case of Interaction Protocols", In Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi Agent Systems, Bologna, Italy, 2002, pp. 960-967.

[9] W. Reisig, Petri Nets: And Introduction, Springer-Verlag, Berlin, 1985.

[10] N. Sadeh, N. D. Hildum, D. Kjenstad, D. and A. Tseng, "MASCOT: an agent-based architecture for coordinated mixed-initiative supply chain planning and scheduling", In Proceeding of Workshop on Agent-Based Decision Support in Managing the Internet-Enabled Supply-Chain, 1999. http://www.damas.ift.ulaval.ca/~moyaux/bibtex/sadeh99mascot.pdf (visited at 29/08/2005)

[11] W. Shen and N. Norrie, "An Agent-Based Approach for Dynamic Manufacturing Scheduling", Working Notes of the Agent-based Manufacturing Workshop, Minneapolis, MN. USA, 1998, pp. 117-128.

[12] K. Sycara, and D. Zeng, "Task-based multi-agent coordination for information gathering", In Craig Knoblock and Alon Levy, editors, Working Notes of the AAAI Spring Symposium Series on Information Gathering from Distributed, Heterogeneous Environments, Stanford, CA, 1995, pp. 154-157.