计算机科学2004Vol.31№.2

一种改进的分布式资源发现算法*)

廖 红 周明天
(电子科技大学计算机学院 成都610054)

摘 要 泛洪算法是分布式网络中的传统资源发现算法,但被应用于移动网络中时,该算法并不能保证所有的结点都能发现其他结点的资源,本文将移动agent和改进的泛洪算法结合,使用移动agent完成网络结点间的资源信息交换,提出了一种新的资源发现算法:双向反馈算法(DDF)。通过对DDF的性能分析证明,移动agent的应用和泛洪算法的改进使DDF比传统的泛洪算法收敛得更快,并能较好地适应移动网络环境。

关键词 分布式算法,资源发现

An Improved Distributed Resource Discovery Algorithm

LIAO Hong ZHOU Ming-Tian
(Department of Computer Science,UESTC,Chengdu 610054)

Abstract The Flooding algorithm is a traditional resource discovery algorithm in disrtributed networks. However,it is shown that the Flooding algorithm do not perform well in ad-hoc mobile networks. In this paper,we present an algorithm called Double-direction-feedback (DDF) based on the Flooding algorithm, which uses mobile agent to transmit the resource messages between the network nodes. We prove that DDF can achieve better results in the performance by good convergence and is adaptive to the ad-hoc mobile network envirement.

Keywords Distributed algorithms,Resource Discovery

1 引言

移动网络环境中,无论是资源提供者或资源访问者都随时可能发生动态变化,资源信息的搜索发现成为分布式移动网络中一个值得研究的问题。通过资源发现,网络主机可以自动发现网上其他主机的资源信息,并动态地将自己的资源信息发布在网上。现有的资源发现方式主要分为资源-目录-请求者的集中模式和分布模式[1]。但是,分布式的资源发现算法如泛洪算法[2],交换算法[3],分布感知算法[4,5]等,在每个结点高度动态变化,没有中心服务器的ad-hoc移动网络[1]中并不能保证所有的结点都能发现其他结点的资源。本文指出了泛洪算法在ad-hoc移动网络中存在的缺陷,将移动agent和改进的泛洪算法结合,提出了一种新的资源发现算法:双向反馈算法。通过分析双向反馈算法的性能指标证明在ad-hoc网络下该算法是可行的。

2 资源发现和资源发现图

2.1 资源发现问题和资源发现算法

一个分布式计算机网络中,通过网络连接和信息交换,彼此发现对方结点所拥有的资源,这个问题称为资源发现问题。使整个网络中所有结点都能了解其他所有结点的资源信息而采用的结点通信方法称为资源发现算法。

2.2 资源发现图[2]

根据文[2],定义名词如下:

定义1(资源发现图) 有向图G(V,E)表示网络中各个结点发现资源的情况,其中网络中的每个主机都是图G中的一个结点v,如果主机u了解主机v上的资源信息,则用有向边(u,v)来表示,即(u,v)∈E。

定义2(了解集) 对任意结点u,Γ(u)是u的了解集,Γ(u)包含了u所了解的所有主机结点,边(u,v)∈E当且仅当v∈Γ(u)。即:(u,v)∈E iff v∈Γ(u)。其中,总有u∈Γ(u)。

定义3(消息) 是一系列结点的集合。结点u可以将消息发向任何一个属于了解集Γ(u)的结点。当结点u收到了来自结点v的消息M,则u的了解集更新为:Γ(u)∪M。在这种情况下,我们称结点u向结点v传送了一个指针集M(pointer set)。

定义4(回合) 网络中每个节点向自己的邻接点发送消息M,并带回反馈消息一次,称为一个回合(round)。

定义5(初始邻接表) 算法开始前,每个结点都被设置了一个初始邻接表(initial list),每个结点只与初始邻接表中的结点进行通信。

定义6(初始邻接图) 将网络中各个结点的初始化邻接表所组成的图称为初始邻接图(initial图)。

从图的角度来说,资源发现的目标在于:对一个有向连通图使用资源发现算法,将这个有向连通图逐步收敛成为一个有向完全图。换句话说,使网络中的每个结点都能了解所有结点的资源信息(包括它自己)。即:对于所有的u∈V,都有Γ(u)=V。

假定网络中的系统时间一定,网络上的各个结点之间的通信是可以进行的,资源发现算法在网络各个结点上并行执行,每个结点执行相同的步骤,以便算法能在有限步骤内实现收敛。讨论算法收敛的前提为:initial图至少是弱连通的。

*)本文的工作受到武器装备预研基金项目(编号:51415010101DZ02)的资助。廖 红 硕士研究生,研究方向:移动网络的资源发现,移动Agent。周明天 博导,教授,研究方向:网络分布式计算,中间件计算。

图1给出了一个例子:结点A向结点B发送消息M,M中包括了它的了解集中的所有信息(包括它自己):(Γ(A)=(A,B,c,d,e)),结点B分析该消息M,将自己的了解集Γ(B)进行更新:Γ'(B)= Γ(B)∪M ,M=Γ(A)。

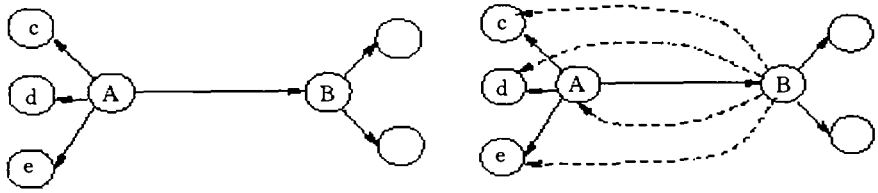

图1 资源发现图

3 泛洪算法及其性能分析

3.1 泛洪算法[2,3]

在泛洪算法的每个回合中,每个结点v按照初始邻接表连接表中的结点,将本结点v的Γ(v)更新的部分(记为Γ(v)^{updates})传送给这些结点,Γ(v)^{updates}是继上次v向该结点发送消息之后更新的部分。收到Γ(v)^{updates}的结点u根据收到的Γ(v)^{updates}相应更新自己的Γ(u),将Γ(u)和Γ(v)^{updates}合并:Γ(u)←Γ(u)∪Γ(v)^{updates},如此反复,直到算法收敛。

3.2 资源发现算法的性能分析指标

为衡量算法的运行时间和网络通信量,作如下定义:

定义7(时间复杂度,time complexity) 算法收敛所需要进行的回合数称为时间复杂度。并非算法收敛所要执行的时间。在算法分析中,假定在下个回合开始前,有充足的时间完成全网所有结点本个回合的操作。

定义8(指针通信复杂度,pointer communication complexity) 在算法收敛的过程中所需传送的结点资源信息总和。一个结点的资源信息算为一个指针。

定义9(连接通信复杂度,connection communication complexity) 在算法收敛的过程中产生的连接总数。当u向v发送一次消息,称为一个连接产生。

3.3 泛洪算法的性能分析

泛洪算法收敛的时间复杂度为初始邻接图的直径:$d^{initial}$。将初始邻接图的边数计为$m^{initial}$,该算法的指针通信复杂度则为$\theta(n * m^{initial})$(n为初始邻接图中结点个数),因为在算法达到收敛的过程中,每个指针都必须在初始邻接图的每条边上传送一次。其连接通信复杂度是$\theta(d^{initial} * m^{initial})$,因为算法收敛共需要$d^{initial}$个回合,每个回合都有$m^{initial}$个连接被打开。

4 泛洪算法的缺陷

若算法开始之前,存在着这样的结点v,没有一个结点将结点v纳入自己预先设置的初始邻接表initial list,则这个资源发现图在算法收敛后将不能演进为一个完全有向图。即结点v在算法收敛后,将无法完全得知其他结点的资源信息。

举例如下:

设结点v的初始邻接表记为I(v)。I(v)=Ad(u)表示结点v只能将自己的Γ(v)^{updates}根据结点u的地址发送给结点u。

设算法开始前,初始邻接图如图2所示。

则如图2中各个结点的初始邻接表为:

I(A)=Ad(B),I(B)=Ad(C),I(C)=Ad(A),I(D)=Ad(B)。

算法开始前,各结点的了解集为:

Γ(A)=(A),Γ(B)=(B),Γ(C)=(C),Γ(D)=(D)。

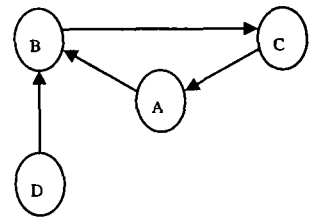

图2 结点的初始邻接图

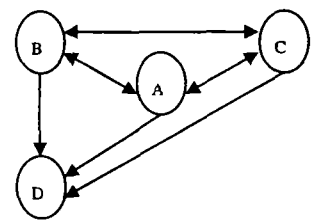

图3 泛洪算法收敛后的资源发现图

在算法开始之前,网络上所有的结点除了本机的资源信息之外,对其他结点的资源一无所知。第一个回合开始,结点A根据自己的初始邻接表,发送Γ(A)^{updates}=(A)给结点B,B将Γ(B)与Γ(A)^{updates}合并:Γ(B)=Γ(B)∪Γ(A)^{updates}=(B,A)。在结点A向B发送消息的同时,B根据自己的初始邻接表,发送Γ(B)^{updates}=(B)给结点C,C将Γ(C)与Γ(B)^{updates}合并:Γ(C)=Γ(C)∪Γ(B)^{updates}=(B,C)。其余类推。到第三个回合结束时,算法收敛。收敛后的资源发现图如图3所示。可见,由于在初始邻接图中,没有任何一个结点将D设为自己的传递结点,导致在算法执行过程中,没有结点向结点D透露其他结点的信息。更进一步地分析,若结点D还负责通知另一个结点E,且结点E只能由结点D来通知,则结点E也被连带无法得知网上的资源情况。D,E的资源情况可被其他结点发现,而D,E自己却无法发现其他资源。

结论:如果在initial图中存在入度为0的结点,则算法收敛后将无法演进为资源发现完全图。(另:initial图中不会出现出度为0的结点,因为每个结点都至少要指定一个initial点。)

在实际的ad-hoc移动网络中,每个结点随机地加入网络资源发现系统,每个结点独立运行。它只能预先指定自己的initial list却无法事先知道自己是否至少在其他一个结点的initial list中。为此,针对ad-hoc移动网络环境,我们提出了一个泛洪算法的改进算法:双向反馈算法。

5 双向反馈算法(DDF)

5.1 算法描述

在每个回合中，网络中的每个结点 v 根据自己的 initial list 发送自己的资源更新集 $\Gamma(v)^{updates}$ 给 initial list 中的每个结点 u，结点 u 收到该信息之后，先反馈自己目前已知的新的资源信息给结点 v，再对自己的 Γ(u) 加以更新；结点 v 根据反馈信息更新自己的了解集 Γ(v)。即网上传递的消息是双向的，带去的是自己所知的新的资源信息，返回的是对方结点上的新的资源信息。可以设定资源的有效时间，结点 u 检查自己的资源信息了解集，在上一个回合后更新的资源信息被反馈回结点 v。在算法实现时，用移动 agent 传递资源更新信息。

对图2所给出的 initial 图，双向反馈算法执行的收敛过程是：

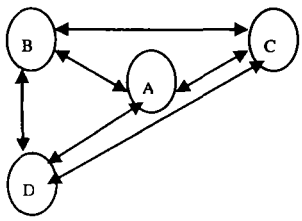

图4　双向反馈算法收敛后的资源发现图

第一回合时，结点 A 发送自己的 $\Gamma(A)^{updates}=(A)$ 给结点 B，结点 B 接到 $\Gamma(A)^{updates}$ 后，先向结点 A 反馈自己的 $\Gamma(B)^{updates}$，然后再根据 $\Gamma(A)^{updates}$ 更新自己的了解集 Γ(B)，Γ(B)=(A,B)。结点 A 接到反馈信息后，更新自己的了解集 Γ(A)，Γ(A)=(A,B)。第二个回合时，算法已经收敛。尽管结点 D 入度为0，还是发现了网上其他结点的资源信息。收敛后的资源发现图如图4所示。

5.2　算法收敛为完全有向图的证明

证明：若 initial 图为弱连通图，且存在入度为0的结点 v，经过有限步双向反馈算法后，该资源发现图总能收敛为有向完全图。

证明：用数学归纳法。

Ⅰ. 当 initial 图中结点个数 n=2时，initial 图中不可能存在入度为0的结点；当 initial 图中结点个数 n=3时，结论显然成立。

Ⅱ. 假设：initial 图中结点个数为 i 时，结论成立。即结点个数为 i 的图必然能收敛为一个完全有向图。则在这 i 个结点的 initial 图中，加一个结点 v，使 i+1个结点的 initial 图也连通。若 v 的 initial list 的结点是原来 i 个结点的图 G_i 中的结点 u，则在第一个回合时，v 连接 u，将自身的资源告知 u，由于 G_i 已为有向完全图，则 v 同时能从 u 处得到其他 i−1个结点的资源信息，即 v 就有 i 条边分别指向其他 i 个结点。又因为 G_i 能在 Round(i) 回合收敛为完全图，则 u 的资源信息将在

（下转第187页）

2004 年网络、通信与信息系统学术会议

征文通知

中国计算机学会西南学会“网络与信息系统专业委员会”、云南省通信学会、四川省计算机学会“网络与信息系统专业委员会”、云南省计算机学会“网络专业委员会”定于 2004 年 7 月 30 日—8 月 2 日联合召开“2004 年网络、通信与信息系统学术会议”，会议由云南大学承办。现将有关征文事宜通知如下，请各学会委员踊跃投稿并积极组织稿件。热诚欢迎全国各地，特别是西南地区的产、学、研同仁们投稿和参会。

1. 征文内容（主要包括但不局限于以下方面）

* 电子政务　* 信息化与西部开发　* 电子商务　* 新型网络和软件体系结构　* 移动通信与移动计算　* 现代通信技术与发展　* 信息安全　* 分布式计算与网格计算

2. 投稿要求

* 论文必须是一次性投稿。通过会议学术组评审被接收的论文将在中文核心期刊 2004 年增刊上发表，论文格式请按《计算机科学》投稿要求撰写，详细要求请咨询《计算机科学》编辑部，电话：（023）63500828，Email：jsjkx@swic.ac.cn；

*稿件一律采用 Word 文档，A4 纸打印件，字数限 5000 字（正文为 5 号宋体，并附软盘）；

*投稿时请务必留下详细通信地址、邮政编码、电话及 Email，以便联系；

*投稿截止时间：2004 年 4 月 15 日；　录用通知时间：2004 年 5 月 15 日。

3. 投稿请寄　610031 成都市西南交通大学计算机与通信工程学院　窦　军副教授收　Email：doujun@tom.com

来研究的重要内容。

参考文献

1 Gronau I, Hartman A, Kirshin A, Nagin K, Olvovsky S. A methodology and architecture for automated software testing. http://www.haifa.il.ibm.com/projects/verification/gtcb/papers/gtcbmanda.pdf, 2000

2 Chow T S. Testing design modeled by finite-state machines. IEEE Transactions on Software Engineering, 1978,4(3): 178～187

3 Offutt J, Abdurazik A. Generating test cases from UML specifications. UML' 99, USA, 1999

4 Poore J H. Introduction to the special issue on: model-based statistical testing of software intensive systems. Information and Software Technology, 2000, 42(12):797～799

5 Beizer B. Black-Box Testing: Techniques for Functional Testing of Software and Systems, Wiley, New York, USA, 1995

6 Jorgensen A, Whittaker J A. An API Testing Method. STAREAST' 00, USA, 2000

7 Fujiwara S, Bochmann G, Khendek F. Test selection based on finite state models. IEEE Transactions on Software Engineering, 1991, 17(6): 591～603

8 Rosaria S, Robinson H. Applying models in your testing process. Information and Software Technology, 2000, 42(12): 815～824

9 Whittaker J A. Stochastic software testing. The Annals of Software Engineering, 1997, 4:115～131

10 Liu C, Richardson D J. Using application states in software testing. ICSE'00, Ireland, 2000

11 Harel D. Statecharts: a visual formalism for complex systems. Science of Computer Programming, 1987, 8(3): 231～274

12 Maurer P M. The design and implementation of a grammar-based data generator. Software Practice & Experience, 1992, 23(3): 223～244

13 Hong H S, Kim Y G, Cha S D. A test sequence selection method for statecharts. The Journal of Software Testing, Verification & Reliability, 2000, 10(4): 203～227

14 Abdurazik A, Offutt J. Using UML collaboration diagrams for static checking and test generation. UML' 00, UK, 2000

15 Peters D K, Parnas D L. Using test oracles generated from program documentation. IEEE Transactions on Software Engineering, 1998, 24(3): 161～173

16 Zhu H, Hall P, May J. Software unit test coverage and adequacy. ACM Computing Surveys, 1997, 29(4): 366～427

17 Sommerville I, Sawyer P. Requirements Engineering: A Good Practice, Wiley, Chichester UK, 1997

18 vritzer A, Larson B. Load testing software using deterministic state testing. ISSTA'93, USA, 1993

19 Avritzer A, Weyuker E J. The automatic generation of load test suites and the assessment of the resulting software. IEEE Transactions on Software Engineering, 1995, 21(9): 705～715

20 Whittaker J A, Thomason M G. A Markov chain model for statistical software testing. IEEE Transactions on Software Engineering, 1994, 20(10): 812～824

21 Walton G H, Poore J H. Generating transition probabilities to support model-based software testing. Software: Practice and Experience, 2000, 30(10): 1095～1106

22 El-Far I K. Automated Construction of Software Behavior Models: [Master's Thesis]. Florida Institute of Technology, Melbourne, Florida, USA ,1999

(上接第44页)

Round(i)个回合后为其他i个结点所知,即有i－1条边分别从除u以外的i－1个结点指向v。此时,G_{i+1}成为一个有向完全图。证毕。

5.3 性能分析

5.3.1 *时间复杂度* 双向反馈算法的时间复杂度为$\Omega(d_{initial})$。在最坏的情况下,即initial图中每个结点的出度仅为1(每个结点只向其他一个结点发送自己所知的资源信息),$d_{initial}$为initial图中最长路径的长度,则$d_{initial}=n-1$。n为initial图中结点的个数。(由于$G_{initial}$是连通的,且每个结点必有出度为1的一条边,因此该连通图要连通n个结点,必须有n－1条边。若有(v－>u),(u－>w),则结点w上的资源信息只有在R_i回合传给u后,才能在R_{i+1}回合传给v。v无法直接从w上得知w的资源信息。因此,算法收敛的回合数就是$d_{initial}$。

5.3.2 *连接通信复杂度* 该算法的连接通信复杂度为$O(m_{initial}*Round)$,上限为$2*m_{initial}*d_{initial}$。在每一个回合中所有结点总共发出$m_{initial}$个连接,在反馈时,每个接受连接的结点又会反馈回一个连接。所以,算法的连接通信上限为$2*m_{initial}*d_{initial}$。

5.3.3 *指针通信复杂度* 其指针通信复杂度为$\Omega(2n*m_{initial})$。因为在达到算法收敛的过程中,每个结点的资源信息都必须在初始邻接图的每条边上发送给对方一次,同时又收回一次对方的指针。收敛过程中共有2n个指针在$m_{initial}$条边上传送。

结束语 针对ad-hoc移动网络环境,我们在现有的泛洪算法的基础上经过改进,将移动agent和改进的泛洪算法相结合,提出了一种新的资源发现算法:双向反馈算法,该方法有两个优点:一是能在未知网络其他结点的initial list的情况下,保证网络中每一个结点都能发现网络其他结点的资源,二是可以节约算法收敛的时间,代价是通信量略有增加。

参考文献

1 王敏毅.面向移动环境的分布对象技术:[博士论文].电子科技大学,2002

2 Harchol-Balter M, Leighton T, Lewin D. Resource discovery in distributed networks. In: 18th Annual ACMSIGACT/SIGOPS Symposium on Principles of Distributed Computing, May1999

3 Law T, Siu K Y. An O(log n)Randomized Resource Discovery Algorithm. In: 14th Intl. Symposium on Distributed Computing, Oct. 2000

4 Jun Y, Boloni L, Palacz K, Dan C. Agent-Based Resource Discovery, Oct. 1999

5 Kutten S, Peleg D. Deterministic distributed resource discovery. In: 19th Annual ACM SIGACT/SIGOPS Symposium on Principles of Distributed Computing, July 2000