一种依据慈善算法的偶图*k*-完全匹配

李毅超，陈 波，周明天

(电子科技大学计算机科学与工程学院 成都 610054)

【摘要】在扩展一种基于内容的负载共享算法的过程中，总结了将初始化负载分布到集群成员服务器的模型和方法，探讨了依据慈善算法进行偶图一对多匹配即*k*-完全匹配的问题。给出了一些应用慈善算法进行偶图匹配的重要实验结果，并对慈善算法存在的问题和在超图等研究领域的潜在应用进行了讨论。

关 键 词 偶图; 一对多匹配; *k*-完全匹配; 慈善算法

中图分类号 TP311.13 文献标识码 A

A Kind of Charity Algorithm for *k*-Perfect Matching of Bipartite Graph

Li Yichao，Chen Bo，Zhou Mingtian

(School of Computer Science and Engineering，UEST of China Chengdu 610054)

Abstract The present paper summarizes the one-to-many matching model of bipartite model and corresponding methodology formalized in the process of extending a content-based load sharing algorithm, LARD. Such a one-to-many matching is referred to as *k*-perfect matching, and a charity algorithm is given as the solution. Some experimental results for the matching based on charity algorithm are then presented, with open issues and potential applicability in the context of hyper graph left as discussions.

Key words bipartite-graph; one-to-many matching; *k*-perfect matching; charity algorithm

传统负载共享策略的潜在性能问题在于大量的缓存达到率，即存在缓存中的内容匹配请求的比率。考虑到缓存技术的巨大优越性，文献[1]提出了基于内容的请求分派，称为地点相关请求派发(Locality-Aware Request Distribution, LARD)；文献[2]提出了类似的称为缓存亲和(Cache Affinity)的方法；文献[3]考察了LARD的局限性，并提出了通过偶图*k*-完全匹配的模型来配置LARD的初始内容配置的解决方案。本文着重讨论该解决方案给出的慈善算法。

1 LARD及其局限性

如文献[1]所述，LARD是一种基于内容的请求分派机制，其目的是获得后端结点上尽可能高的缓存达到率。传统LARD的一个基本假设是所有的后端结点都能对任意的目标提供服务，意味着在后端所有成员服务器上放置内容的同构性。另一方面，尽管传统的LARD只处理了在负载共享时的缓存亲和问题，也提到了使用基于内容策略时的两个优点：1) 通过划分大数据库到不同后端结点获得更高的二级存储可伸缩性；2) 使某些后端结点成为专门处理特定类型请求的专用服务器，如视频服务器。LARD要实现这两个方面的优点，必须适应异构的内容分布环境，即后端服务器没有提供相同的服务内容集合。在同构情况下，哪个

收稿日期：2003 – 01 – 20

作者简介：李毅超(1969 –)，男，副教授，主要从事计算机网络与通信、网络安全方面的研究.

主机为哪种具体类型的请求服务是通过只依赖于后端服务器负载状态的随机选择确定的。但是这种派发过程中的分配在异构情况下可能出现严重超载问题，因为此时这种选择依赖于服务器负载的状况和内容的物理位置分布。这种矛盾要求给出一种对各种内容类型在后端服务器上合理的初始划分。

2　一对多匹配模型

2.1　k-完全匹配

表1是任意给出的15种内容类型在服务器A-E上的分布，传统的LARD假设所有服务器上都有所有的服务内容类型。但异构环境下，每种服务内容可能只存在于某些特定的服务器上，如表1所示。为陈述简单，本文将使用该假设异构环境作为基本实例，称为HYPO。它也可以采用偶图的形式作为更具可视性的表示，即令表1为一个偶图$G=(U, V, E)$的邻接表的一种变体，U表示主机集合，V表示内容类型集合，当且仅当内容v存在于主机u上时存在边$e=(u,v)\in E$。

表1　一种假设的内容异构分布

内容类型	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
内容分布的后端服务器	A	A	B	B	C	D	B	A	A	B	B	A	A	B	D
	B	B	E	D		E	C	B	C	C	C	B		E	E
	D	C					E	E	D	D	E	C			
		D							E	E					
		E													

这样，初始分配内容类型到它的服务主机上的问题，就转化成为如何选择偶图的连杆来形成一个生成子图，使得每个表示内容的顶点只能具有度数1。该问题理想的解是尽可能相等的平均分配内容类型到每个代表服务器的顶点。定义k-完全匹配(k-Perfect Matching, k-PM)为该问题的一个特例，此时内容类型的数量恰好被主机数量整除，这样每个表示内容类型的顶点度数为1，而每个主机顶点度数为k。推广到不能整除的情况，k的一般定义为 $k=\left[\frac{|\text{contents}|}{|\text{hosts}|}\right]=\left[\frac{|U|}{|V|}\right]$，这样在HYPO中，$k=\left[\frac{15}{5}\right]=3$。

2.2　完美对集—— 特例

在一个偶图$G=(U,V,E)$中，当$|U|=|V|$成立，则k=1，这样k-完全匹配问题退化成经典的完美对集问题以及相应的最大对集问题。解决偶图对集问题的一系列算法，主要是基于从用贪心算法得到的一个初始匹配M中寻找可增扩路来实现。这方面工作的主要代表有文献[4～6]等所述及的匈牙利方法，这些方法不仅解决了普通的对集问题，还解决了加权图的最优匹配问题。偶图对集算法大量变体的主要差别在于使用搜寻可增扩路的具体细节，例如广度优先搜索(Breadth First Search, BFS)，深度优先搜索(Depth First Search, DFS)，Push Relabel 以及(Alt, Blum, Mehlhorm and Paul) ABMP方法[7, 8]。作为k-PM的一个特例，对集问题表示了把n种内容类型分配到恰好n个后端服务器上的情况。用上面提到的算法，可以在存在完美对集的情况下得到一个解。完美对集的存在判定基于霍氏定理(Hall's Theorem)，即一个偶图$G(U,V,E)$具有完美匹配，当且仅当对每个$X\subseteq U$及$Y\subseteq V$，有$|G_G(X)|\geq|X|$且$|G_G(Y)|\geq|Y|$。当霍氏定理条件不能满足时，没有办法把每个内容类型都分配到一个空闲的服务器上，此时先利用上面给出的算法先获得最大匹配。代表未分配内容的顶点和所有的主机可以构成一个G的偶图子图$G'=(U,V',E')$，V'是未分配内容顶点集，E'是连接U和V'中顶点的边集。在该图上递归寻找新的最大匹配，直到不存在未分配内容。

2.3　3种解决k-PM的策略

1) 递归法：递归法与在2.2节中为不满足霍氏完美对集存在条件的偶图匹配的情形相似。该递归过程如下：(1) 对偶图$G(U,V,E)$，使用最大对集算法寻找饱和主机结点集合U的最大匹配M，令$V':=V-V_A$，V_A表示M中的内容顶点，令$E'\subset E$为关联顶点集V'的边集；(2) 如果$V'=F$，方法中止，否则构造一个G的子图G'，使其具有顶点集$U\bigcup V'$和边集E'；(3) 令$G:=G'$，重复(1)～(3)。递归算法最重要的是要确保其在有限次递归后停止，由于是一个非空偶图，每次递归中偶图中最少存在一个大小为1的对集，即存在一条从U到V的连杆，这样，最坏的情况是经过$|V|$次循环，所有的内容都分配到同一个服务器上；最好的情况是每次循环都恰好存在大小为$|U|$的匹配，方法会在 k 步递归停止。

2) 复制法：从本质上而言，一对一分配问题和一对多分配问题的关键差别在于偶图的一部U的集合大小比另外一部小，但是却又必须将数量多的一部全部匹配到数量少的一部，导致了传统的一对一匹配方法

无效。因此，可以将原始偶图中的一个主机顶点画成冗余的k个，使得两部的顶点数量相等，这种方法尤其对内容顶点数量恰好被主机顶点数量整除为k的情况适用。这种方法的可行性在于，它实际上可以看作把每个服务器划分成了k个部分，而分配内容的意义在于均匀地把所有内容类型分配到所有的这些部分上去，这就转化成2.2节的情形。虽然这种方法比较直观，但是大大地扩张了一个图的顶点数和边数，从而导致更大的空间复杂度及寻找可增扩路的搜索空间。

3) 慈善法：称某个主机是k-饱和的，是指已经有k个内容顶点被匹配到该主机顶点。仔细观察k-PM问题，成功匹配的关键在于避免出现下面的情况，即当一个内容顶点除了被匹配到已经k-饱和的主机顶点上外别无选择，然而另一方面，在使得该被选主机k-饱和的k个内容顶点中有一些实际上可以被转移到其他尚未k-饱和的结点上。在慈善活动中，有两个称为慈善准则(Charity Principles)的基本原则：(1) 总是尽力先帮助生存机会最少的弱势者；(2) 对一个人的帮助总是限制在一定范围内，以够用为原则，以便用有限的资源能够帮助尽可能多的人。基于这种基本思想，在文献[3]中提出了慈善算法(Charity Algorithm, CA)。

3 慈善算法

3.1 基本量度定义

定义 1 适应度(Adaptability Degree, ADD)是V中一个具体内容顶点的度数，它可用于描述该结点的选择机会数量，以便确定哪个内容最需要先分配。

定义 2 饱和边界(Saturation Bound, SB)是大于等于内容顶点数$|V|$与服务器顶点数$|U|$比率的最小整数，与k-PM中参数k的定义一致。该度量值用于确定给定主机顶点容量的上界。

定义 3 可用度(Availability Degree, AVD)是一个主机顶点的饱和边界减去已经分配到该主机上内容数量的值，该度量值表示了在分配过程中当前的可用容量。

定义 4 多样度(Versatility Degree, VD)是通过边连接到一个主机顶点的所有尚未分配的邻接内容顶点数目，该度量值表示特定主机对未来不同类型请求的服务能力。

定义 5 倾向度(Amicability Degree, AMD)是给定主机顶点可用度与多样度之间的比率，该度量值在慈善算法中用来确定把当前内容指派到哪个主机。

3.2 算法描述

第一步：初始化。对U中所有主机顶点计算饱和边界SB，初始化每个主机顶点的可用度AVD为该结点的饱和边界SB值，多样度VD为该顶点的度数。通过AVD与VD之间的比值可以得到它们的初始倾向度。初始化V中每个顶点的适应度ADD为它们在图中的度数。

表2 在HYPO上进行慈善算法演示

	ADD	A	B	C	D	E	
lnit		3/6	3/10	3/7	3/7	3/10	
5	1			2/6			C
13	1	2/5					A
3	2		2/9			3/9	B
4	2		2/8		2/6		D
6	2				2/5	2/8	E
14	2					2/7	B
15	2				1/4	2/6	D
1	3	1/4	1/6		1/3		A
7	3		1/5	1/5		2/5	C
8	3	1/3	1/4			1/4	E
11	3		0/3	1/4		1/3	B
12	3	0/2	0/2	1/3			A
9	4	0/1		0/2	1/2	1/2	C
10	4		0/1	0/1	0/1	1/1	D
2	5	0/0	0/0	0/0	0/0	0/0	E

第二步：递归开始。根据内容顶点的适应度ADD值升序输入V中顶点，如果其中两个顶点的ADD值相同，任意顺序都可接受。

第三步：对当前输入顶点进行分配。从主机顶点集合U中选择具有最大倾向度AMD值的顶点来匹配输入的内容顶点。当存在多个主机顶点具有相同AMD值时，应用如下规则：1) 选择具有最大可用度的AVD的顶点；2) 当AVD仍然相等时，从中任意选择一个输出匹配的顶点对。

第四步：修改度量值。一旦第三步成功，修改主机结点度量值，包括：1) 对在第三步中选择了的主机顶点，递减其可用度AVD值；2) 把第三步输入的内容顶点所有邻接主机顶点的多样度VD减1，并获得下次递归使用的新倾向度AMD值。

第五步：从第二步开始递归。递归的成功停止条件是所有的内容顶点被分配。

使用慈善算法处理HYPO的样例如表2所示。

4 实验结果

实验集的目的是通过任意产生的偶图验证慈善算法CA的有效性，同时获取一些时间标度。在实验集中，我们使用稀疏矩阵表示偶图，其中的非零元素表示连杆。顶点集U和V的数量以及稀疏度是随机指定的以确保相应的图不会局限在某个狭窄的类型。在慈善算法中对内容顶点度数的排序采用快速排序算法。实验用JDK 1.3.1在Microsoft Windows 2000 Professional上实现，机器配置为Intel Pentium III 733 MHz/512 M。表3给出了对超过100 000个随机偶图处理的部分结果。

表3　部分实验结果

$\|U\|$	$\|V\|$	稀疏度	$\|E\|$	孤立顶点	排序(S)	匹配数	未匹配数	错误	循环次数	匹配时间/s
340	901	0.740 7	226 389	0	0.01	901	0	0	226 389	0.120
298	1 752	0.553 9	287 126	0	0	1 752	0	0	287 126	0.111
114	1 301	0.608 9	901 36	0	0	1 301	0	0	901 36	0.030
20	4 761	0.856 6	31 208	0	0.01	4 751	0	0	31 208	0.050
28	6 071	0.659 4	112 152	0	0.02	6 071	0	0	112 162	0.070
179	2 651	0.044 6	211 40	3	0.01	2 648	3	0	21 140	0.030
16	1 017	0.952 6	15 490	0	0	1 017	0	0	15 490	0
31	9 427	0.421 3	123 456	0	0.01	9 427	0	0	123 456	0.280
342	1 286	0.071 4	31 682	0	0	1 286	0	0	31 682	0.030
115	933	0.054 4	5 897	0	0	933	0	0	5 897	0.010
811	1 072	0.073 4	63 616	0	0	1 072	0	0	63 616	0.020
90	1 408	0.643 4	31 697	0	0	1 408	0	0	31 697	0.030
73	2 179	0.003 0	449	1 772	0	407	1 772	0	449	0
86	936	0.985 0	79 211	0	0	936	0	0	79 211	0.050
87	7 304	0.533 9	339 846	0	0.01	7 304	0	0	339 846	0.180
221	1 698	0.858 0	322 211	0	0	1 698	0	0	322 211	0.101
10	1 760	0.054 2	1 006	986	0	774	986	0	1 006	0

5 结 论

本文对慈善算法进行一个粗略的计算复杂度分析，由于匹配一个内容顶点v_i 的复杂度是$O(valence(v_i))$，则全局匹配复杂度是V中每个顶点v_i 复杂度之和，即 $sum(valence(v_i))=|E|$。因此CA的时间复杂度是$O(|E|)$。

关于偶图一对多匹配问题以及慈善算法需要进一步研究的问题包括：1) 偶图k-PM存在性判定的充分必要条件问题；2) 通过使用超图的超边表示主机，使用超图顶点表示内容类型，一种k-PM问题的超图形式可以得到，为此希望超图研究的理论成果与一对多匹配的研究结果能相互丰富。3) 与经典对集问题相似，加权偶图是一种重要的模型，可以表征在匹配中内容与主机之间的其他因素的亲和度，对加权偶图最佳k-PM的求解，将可望有重要价值。

参 考 文 献

[1] Pai V S. Locality-aware request distribution in cluster-based network servers[C]. In: Proceedings of the Eighth International Conference on Architectural Support for Programming Languages and Operating Systems (VIII), San Jose, CA, 1998. 205-216

[2] Bunt R B, Eager D L, Oster G M, *et al*. Achieving load balance and effective caching in clustered web servers[C]. In: Proceedings of the Fourth International Web Caching Workshop, San Diego, California, 1999. 159-169

[3] 陈　波，周明天，An extension to LARD with *k*-perfect matching of bipartite graph[C]. 武汉：华中科技大学出版社，2002. 11-19

[4] Edmonds J. Paths, trees and flowers[J]. Canada. J. Math., 1965, 17: 449-67

[5] Kuhn H W. The hungarian method for the assignment problem[J]. Naval Res. Logist, 1955, (2): 83-97

[6] Munkres J. Algorithms for the assignment and transportation problems[J]. Soc. Indust. Appl. Math., 1957, 51: 32-38

[7] Goldberg A V. Tarjan R E. A new approach to the maximum-flow problem[J]. Assoc. Comput. Mach., 1988, 35(4):921-940

[8] Alt H, Blum N, Mehlhorn K, *et al*. Computing a maximum cardinality matching in a bipartite graph in time [J]. Inform. Process. Lett., 1991, 37: 237-240

编　辑　熊思亮