软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(3):662−676 [doi: 10.3724/SP.J.1001.2012.03990]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于语义吸引的节点规模估计方法*

马行空+, 王意洁, 郑 重

(国防科学技术大学 计算机学院 并行与分布处理国家重点实验室,湖南 长沙 410073)

Network Size Estimation Method Based Semantic Attraction

MA Xing-Kong+, WANG Yi-Jie, ZHENG Zhong

(National Key Laboratory for Parallel and Distributed Processing, College of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: mxkong@gmail.com

Ma XK, Wang YJ, Zheng Z. Network size estimation method based semantic attraction. *Journal of Software*, 2012,23(3):662−676. http://www.jos.org.cn/1000-9825/3990.htm

Abstract: Network size is the fundamental information of the distributed applications. Network size estimation methods must feature both high accuracy and adequate robustness in order to adapt to a large environment with a high node churn. Considering the fact that the existing network size estimation methods mainly focus on single optimization objective and fail to ensure accuracy and robustness simultaneously, a network size estimation method based semantic attraction—SEBSA is proposed in this paper. As the semantic information in SEBSA, hash values are hashed in real intervals by the peers' identifies. The peers with adjacent hash values in SEBSA periodically exchange hash neighbors to attract the most adjacent peers in a hash space quickly. Meanwhile, every peer computes the average spacing among hash values of the hash neighbors to estimate network size. Theoretic analysis and experimental results reveal that compared with existing size estimation methods, SEBSA can provide accurate size estimation information quickly even in continually fluctuating network environment.

Key words: network size; size estimation; semantic attraction; aggregation estimation; P2P

摘 要: 节点规模是各种分布式应用的基础信息,节点波动的大规模网络环境要求节点规模估计方法具有较高的估计精度和较强的鲁棒性,已有的节点规模估计方法多侧重于某个方面的优化而未能充分权衡计算精度和鲁棒性.提出一种基于语义吸引的节点规模估计方法——SEBSA(a network size estimation method based semantic attraction).SEBSA 将每个节点标识所对应的实数区间上的哈希值作为语义信息,节点通过与哈希值临近的节点周期性地交换哈希空间上的邻居信息,以快速吸引与自己哈希值最近的一组节点,测量该组节点哈希值的平均间距以估计节点规模.理论分析和实验结果表明,相对于已有方法,SEBSA 在节点频繁波动的网络环境中仍然能够快速地提供准确的节点规模信息.

关键词: 网络规模;规模估计;语义吸引;聚集值估计;P2P

* 基金项目: 国家自然科学基金(60873215); 国家重点基础研究发展计划(973)(2011CB302601); 湖南省自然科学杰出青年基金(S2010J5050); 高等学校博士学科点专项科研基金(200899980003)

收稿时间: 2010-05-30; 定稿时间: 2011-01-20

CNKI 网络优先出版: 2011-05-26 13:48, http://www.cnki.net/kcms/detail/11.2560.tp.20110526.1348.005.html

中图法分类号: TP393 文献标识码: A

当前的分布式系统以其高扩展性、自组织和容错性而得到广泛应用,然而在大规模网络中,由于系统的全分布式特点和动态性,使得对系统的控制和检测变得很困难,而分布式系统需要节点规模信息为系统控制和检测提供有效的支持.例如:在资源定位系统 Pastry[1]和 Chord[2]中,为了提高覆盖网中路由匹配的准确度,同时又不至于路由匹配开销过大,可根据节点规模信息确定节点的路由表大小;为了设置合理的查询超时时间,也可根据节点规模信息确定最大的路由跳数;在分布式成员管理协议 SCAMP[3]中,为了保证可靠的数据传输,同时又不至于数据冗余过大,可根据节点规模信息确定节点的邻居列表大小;在发布订阅系统 TERA[4]中,为了保持不同订阅主题被转发概率的均衡性,可根据订阅主题的流行度确定是否将订阅加入节点的访问点查询列表中,而订阅主题的流行度则根据相应簇规模大小确定.

在大规模动态网络环境中,节点规模时时都在变化,而规模估计过程需要耗费一定的时间,因此很难精确地获得节点规模大小.另外,网络的动态性也决定了规模估计的方法需要周期性运行以适应节点规模的变化.这就要求所设计的规模估计方法同时具有较高的计算精度和较强的鲁棒性.而以往的规模估计方法大多侧重于某个方面的优化,没有对规模估计精度和鲁棒性进行权衡考虑.

本文提出一种基于语义吸引的节点规模估计方法(a network size estimation method based semantic attraction,简称 SEBSA).SEBSA 将每个节点标识对应到实数空间上的一个哈希值,每个节点将该哈希值作为语义信息,在哈希空间上临近的节点相互交换哈希邻居信息以吸引与自己在哈希空间上最近的一组节点,最终测量该组节点哈希值的平均间距以估计节点规模.相对于已有规模估计方法,SEBSA 在节点频繁波动的网络环境中仍然能够快速地提供准确的节点规模信息.

1 相关工作

早期的节点规模估计方法主要考虑计算精度和计算延迟,即能够在短时间内得到精度较高的计算结果.典型的如反熵聚集方法(anti-entropy aggregation protocol,简称 AAP)[5]、增长随机行走方法(random increasing walk,简称 RIW)[6]以及 FM 方法[7,8].AAP 方法通过节点之间不断反熵求平均值,在静态环境下,能够在 40 轮左右的时间内使各个节点的规模估计值收敛到实际的节点规模,但是对于链路丢包、节点频繁波动的动态网络环境的适应能力有限.而 RIW 方法则根据节点标识大小进行随机行走,将消息转发给标识更大的节点以达到快速收敛的目的,但是该方法要求所有节点从 1~N 进行编号,这在动态网络环境下是难以实现的.另外,FM 方法依指数概率为每个节点分配标识,标识为 0 的概率为 1/2,标识为 1 的概率为 1/4,依此类推,节点周期性交换标识,最终根据收集到的连续标识的个数确定节点规模.该方法具有较小的计算延迟,但是在节点频繁加入退出和大量节点失效时,自适应效果不够理想.Kennedy 等人[8]提出 Count-Sketch-Reset 方法,其思想是,将 FM 方法中的标识位向量修改为整数向量,以更好地适应网络的动态性,但该方法较高的计算精度是以大量的通信开销为代价.

为了提高对网络动态性的适应能力,抽样碰撞方法(sample and collide,简称 SC)[9]、跳步采样方法(hop sampling,简称 HS)[10]、区间密度方法(interval density approach,简称 IDA)[10]、极值传播方法(extrema propagation,简称 EP)[11]以及环形结构化覆盖网下的规模估计方法[12]等被相继提出.SC 方法是由初始节点发起随机行走过程,直到一个已被选中的节点再次被选中为止.该方法可以有效地应对节点和链路的波动性,但在大规模网络环境下的计算延迟较长.HS 方法由初始节点广播一条消息,收到该消息的节点以概率的方式返回给初始节点,其概率的大小基于节点距离初始点的距离.该方法在网络直径较小的覆盖网下具有较快的收敛性,但由发起节点引起的消息泛洪是不可忽略的.另外,IDA 方法通过统计要采样区间的节点数目以估计节点规模大小,通过节点间发送心跳消息以采样节点和发现邻居节点是否失效,然而对于有限的邻居列表,统计特定区间内所有节点的计算延迟是难以容忍的.EP 方法中,每个节点根据已知的概率分布函数生成一个随机数,保留所有随机数的最小值,节点周期性的生成随机数可获取多个最小值,该组最小值构成的向量与节点规模相关,该方法需要大量的最小值采样结果才能达到较高的计算精度.Shafaat 等人[12]提出了针对环形结构化覆盖网的规模估计方法,其基

本思想是,每个节点通过随机行走的方式获取相邻节点间的平均间距,进而根据总区间长度与平均间距的比值以确定节点规模.该方法仅对环形结构化覆盖网有效,应用场景受限.另外,该方法采用不断重启的方式以应对网络动态性,极大增加了通信开销.

以上方法大多侧重于某个方面的优化,并没有在计算精度和鲁棒性之间有效地权衡考虑.而在动态网络环境下,节点加入退出、节点失效、丢包等情况频繁发生,因此,如何在保证计算精度的前提下提高方法的鲁棒性就显得格外重要.SEBSA 根据距离节点自身最近的一组节点的哈希值以计算节点规模估计值,节点的哈希值作为语义信息,语义吸引体现在哈希值临近的节点相互交换哈希空间上的邻居信息,保证了每个节点在哈希空间上可以快速找到距离自己最近的一组哈希值,通过采样多组节点的规模估计值保证了计算精度,周期性地交换哈希空间上的邻居信息保证了在动态环境下的鲁棒性.

2 SEBSA 基本思想

SEBSA 采用 MD-5 算法将每个节点标识映射到(0,1)区间上的一个哈希值并达到均匀分布效果,其基本思想是,每个节点首先通过邻居维护协议获取与自身哈希值距离最近的一组节点,然后计算该组节点哈希值的平均间距以得到节点规模估计值,最后通过邻居维护协议采样多组节点规模估计值并求解其平均值以提高计算精度.

2.1 邻居维护

SEBSA 的邻居维护分为覆盖网邻居维护协议(overlay neighbor management protocol,简称 ONMP)和哈希邻居维护协议(Hash neighbor management protocol,简称 HNMP).覆盖网邻居维护协议通过节点邻居交换以保证覆盖网的高连通性和鲁棒性,同时可以通过邻居采样规模估计值提高计算精度;哈希邻居维护协议中,节点周期性更新哈希邻居表,根据哈希邻居表中相邻哈希值的平均间距估计节点规模.

覆盖网邻居维护协议 ONMP 采用 CYCLON[14]维护节点的邻居列表(neighbor list,简称 NL),其基本思想是,每个节点周期性地从邻居列表中随机选择一个邻居交换各自的邻居信息,其交换后的节点度分布近似为泊松分布,覆盖网拓扑近似为随机图,具有很高的节点连通性、鲁棒性和较小的网络直径,如图 1 所示.

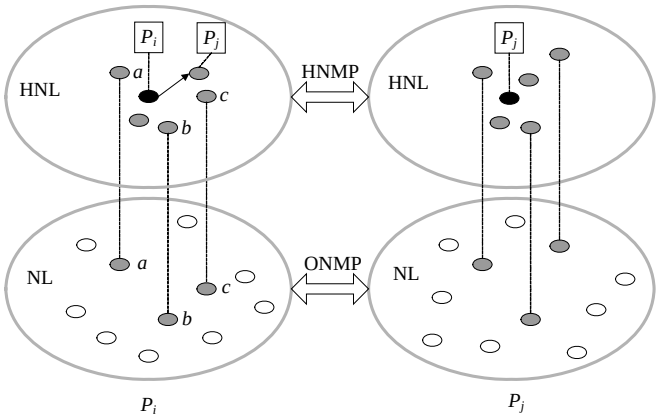


Fig.1 Neighbor management protocols of SEBSA

图 1 SEBSA 邻居维护协议

哈希邻居维护协议 HNMP 中,每个节点维护一个哈希邻居表(Hash neighbor list,简称 HNL),该哈希邻居表代表了与节点哈希值距离最近的一组节点集合.对于任意节点 P_i,该协议采用两种方式更新节点的哈希邻居表:一是节点周期性的从覆盖网邻居维护协议采样邻居的节点标识,并与已有的哈希邻居表中节点的标识对比,选择距离自身节点标识更近的节点加入到哈希邻居表中;二是从节点 P_i 的哈希邻居表中随机选择一个节点 P_j,双方交换其哈希邻居表,对于长度为 L 的哈希邻居表来说,P_i 和 P_j 只保留距离各自节点标识最近的 L 个节点.如图

1 所示,节点 P_i 既从覆盖网邻居维护协议的邻居列表中获取节点 a,b,c 以更新哈希邻居表,另外从哈希邻居表中选择了节点 P_j 并相互交换哈希邻居表.

2.2 规模估计

SEBSA 利用邻居维护协议将距离自身节点标识最近的节点集合加入到哈希邻居表中,然后计算哈希邻居表中相邻节点标识间的平均间距以估计节点规模.其语义吸引就体现在通过与当前哈希邻居表中的节点交换各自的哈希邻居表,快速吸收距离自己更近的节点.

具体来说,每个节点的哈希邻居表通过迭代更新,最终获得系统中与自身标识距离最近的节点集合并达到稳定状态.其中,节点自身的哈希值也属于该列表.设该节点集合的长度为 L,且该集合中节点标识对应哈希值的最大值和最小值分别为 $X_{\max},X_{\min}$,则节点对系统中哈希值的平均间距的估计为$(X_{\max}-X_{\min})/(L-1)$.由于所有节点哈希值满足(0,1)区间上的均匀分布,则节点规模估计值为 $v_i=(L-1)/(X_{\max}-X_{\min})$.同时,覆盖网邻居维护协议中相互交换邻居的节点规模估计值,通过计算多个节点规模估计值的平均值可以有效提高规模估计精度.

3 SEBSA 描述

SEBSA 能够准确快速估计节点规模的关键是保证哈希邻居表 HNL 在动态网络环境下快速找到距离节点自身最近的一组哈希值,这就需要根据节点可能处于的状态采用不同的策略以更新哈希邻居表.

具体来说,在节点加入时,SEBSA 采用随机行走的方式从随机行走路径上选择距离新节点标识最近的节点加入到新节点的 HNL 中;节点正常运行时,SEBSA 通过从 NL 中采样和从 HNL 中随机选择节点交换双方的 HNL 这两种方式以更新 HNL;为了处理 HNL 中的失效节点,SEBSA 周期性交换 HNL 可以判断对方节点是否失效,同时交换各自已知的失效节点信息,从而在 HNL 中删除已失效的节点.最后,当更新节点规模估计值时,在不增加通信开销的前提下收集多个邻居节点的规模估计值以提高计算精度.

3.1 节点加入处理

新节点加入时,首先通过 MD-5 哈希函数获取一个节点标识,然后构建覆盖网邻居维护协议的邻居列表 NL 和哈希邻居维护协议的哈希邻居表 HNL.为了保证邻居列表 NL 中邻居选择的均匀性,采用并发随机行走的方式从系统中选择邻居,即对于每个随机行走过程设置一个随机行走长度 TTL,从系统的共知节点中随机选择一个邻居作为下一跳目的节点,再由该节点从其邻居列表中随机选择一个节点作为下一跳目的节点,直到跳数达到 *TTL* 值为止.此时,所到达的节点就作为新节点的一个邻居.

对于哈希邻居表 HNL,为了在初始时获取距离节点自身标识最近的节点集合,在上述随机行走过程中,同时从下一跳节点的邻居列表和哈希邻居表中选择一个距离新节点标识最近的节点,并将该节点标识值和节点号继续往下发送.当跳数到达 *TTL* 值时,选择该随机过程中距离新节点标识最近的节点作为哈希邻居表中的一项.其构建邻居列表和哈希邻居表的算法如图 2 所示.

```
//newNode:新节点                                  introducer:入口节点,新节点通过其加入 SEBSA
//nearestNode:当前距离新节点标识最近的节点        joinTTL:随机行走的长度
//currentWalkLength:当前随机行走的长度            C:节点的邻居列表 NL 的大小
//currentWalkIndex:当前随机行走的数目
1. 从 introducer 的邻居列表 NL 中随机选择一个节点 walker,currentWalkLength=0 且 nearestNode=walker;
2. 如果 CurrentWalkLength<joinTTL,则
     2.1. 从 walker.NL 中随机选择一个节点 newWalker,且 walker=newWalker;
     2.2. 将 walker.NL 和 walker.HNL 的所有节点的哈希值与 nearestNode 的哈希值对比,找到距离 newNode
          最近的节点 P,且 nearestNode=P;
     2.3. CurrentWalkLength++,转向步骤 2;
3. 如果 CurrentWalkLength>=joinTTL,则将 walker 加入到 newNode.NL 中,将 nearestNode 加入到 newNode 中;
4. currentWalkIndex++,如果 currentWalkIndex<C,转向步骤 1.
```

Fig.2 Algorithm of building NL and HNL of the new nodes

图 2 新节点的邻居列表 NL 和哈希邻居表 HNL 构建算法

3.2 哈希邻居表更新

哈希邻居表的长度假设为 L,SEBSA 采用两种方式更新哈希邻居表 HNL:一是源节点从覆盖网邻居维护协议的邻居列表 NL 中采样,获取 NL 和 HNL 中距离源节点标识最近的 L 个节点作为源节点的 HNL.另一种是周期性地从 HNL 中随机选择一个节点,双方交换各自的 HNL,每个节点从自己的 HNL 和对方的 HNL 中选择距离自身节点标识最近的 L 个节点作为新的 HNL.其算法实现过程如图 3 所示.

```
//myNode:节点本身                hashID:接点的哈希值              L:哈希邻居表的大小
(a) 通过采样 myNode.NL 更新 myNode.HNL
1. 将 myNode.NL 和 myNode.HNL 的节点合并加入到一个集合 S 中;
2. S 中所有节点根据与 myNode.hashID 的距离的绝对值由小到大排序;
3. 取前 L 个节点作为新的 myNode.HNL.
(b) 通过其他节点的哈希邻居表更新 myNode.HNL
1. 从 myNode.HNL 中随机选择一个节点 randomNode,并将 myNode.HNL 发送给 randomNode;
2. randomNode 收到 myNode.HNL,将 randomNode.HNL 发送给 myNode,然后合并 muNode.HNL 和
   randomNode.HNL,并按照与 randomNode.hashID 的距离的绝对值由小到大排序,取前 L 个作为新的
   randomNode.HNL;
3. myNode 收到 randomNode.HNL,合并 myNode.HNL 和 randomNode.HNL,按照与 myNode.hashID 的距离的绝对值
   由小到大排序,取前 L 个作为新的 randomNode.HNL.
```

Fig.3 Algorithm of updating HNL of SEBSA

图 3 哈希邻居表更新算法

3.3 节点失效处理

覆盖网邻居维护协议的失效检测采用 CYCLON 方法,即通过在邻居交换时保留年龄小的节点以去除长时间没有更新的节点.

哈希邻居维护协议中节点失效处理的方法是:在如图 3(b)所示的实现过程中,若源节点所选择的节点没有应答,则认为该节点失效,并将其从源节点的哈希邻居表中删除,同时将该失效节点的标识存放在源节点的失效节点过滤器(failed node bloomfilter,简称 FNB)当中;如果被选择节点收到源节点的消息,则在交换 HNL 的同时也交换双方的 FNB.每个节点周期性地遍历 HNL 中每个节点是否在其 FNB 中,若被包含在内,则删除;另外,由于每个节点的 FNB 容量有限,因此需要周期性地清空 FNB.

由于哈希值接近的节点交换 FNB 时,对方节点提供的失效节点信息会以很大的概率出现在自己的 HNL 中,因此可以达到快速识别失效节点的目的.另外,上述哈希邻居维护协议的节点失效处理方法是在图 3(b)所示的 HNL 更新算法中捎带交换双方的 FNB,因此不会引起额外的通信开销,算法实现如图 4 所示.

```
//myNode:节点本身                hashID:节点的哈希值              C:邻居列表 NL 的大小
//timeout:节点建立连接的超时时间  sysTime:系统时间
//failNodeBF:失效节点过滤器      clearTime:清空 failNodeBF 的时间间隔
(a) CYCLON 中的节点退出和失效处理算法
1. myNode 收到邻居节点 randomNode.NL 后,合并 myNode.NL 和 randomNode.NL 为 S;
2. 节点的年龄 age 代表了节点的新鲜程度,S 中的节点按照年龄从小到大排序,S 中前 C 个节点作为新的 myNode.NL.
(b) 哈希邻居表的维护协议中的节点退出和失效处理算法
1. myNode 从 myNode.HNL 中随机选择一个节点 randomNode,并向 randomNode 发送 myNode.failNodeBF,
   且 myNode 启动定时器 timer;
2. 若 randomNode 收到 myNode 消息后,将 randomNode.failNodeBF 发送给 myNode,然后将 myNode.failNodeBF
   与 randomNode.failNodeBF 合并,作为新的 randomNode.failNodeBF;
3. myNode 收到 randomNode 的消息后,将 myNode.failNodeBF 与 randomNode.failNodeBF 合并,
   作为新的 myNode.failNodeBF,且设置 myNode.timer=0;
4. 若 myNode.timer>timeout,将 randomNode.hashID 加入到 myNode.failNodeBF 中,
   并将 randomNode 从 myNode.HNL 中删除;
5. 若 sysTime%clearTime==0,则清空 myNode.failNodeBF.
```

Fig.4 Algorithm of process failure nodes

图 4 节点失效处理算法

3.4 规模估计值更新

为了提高规模估计值的估计精度,在覆盖网邻居维护协议的邻居列表更新时,将邻居的节点规模估计值也发送过来,将收集到的多个节点的规模估计值的均值作为新的规模估计值.收集邻居节点规模估计值的过程在理论上呈指数级增长,因此经过较少轮次的采样迭代后,每个节点的规模估计值可以达到较高的计算精度.另外,由于节点的动态性,每个节点哈希邻居表也在不断变化,因此需要周期性地根据其哈希邻居表重新计算节点规模估计值,其基本过程如图 5 所示.

$//v_i$:节点 P_i 的规模估计值　　L:节点 P_i 的 HNL 的大小　　C:P_i 的 NL 的大小
$//X_{max}$:节点 P_i 的 HNL 中最大节点标识值　　X_{min}:节点 P_i 的 HNL 中最小节点标识值
//CurrentSampleTime:当前采样时间　　SamplePeirod: 采样迭代周期
1. 从覆盖网邻居维护协议的邻居列表 NL 中获取各个邻居的规模估计值 v_j,$1\leqslant j\leqslant C$;
2. 更新节点 P_i 的规模估计值 $v_i=(v_i+\sum_{j=1}^{C}v_j)\big/(C+1)$,CurrentSampleTime++;
3. 如果 CurrentSampleTime>SamplePeirod,$v_i=(L-1)/(X_{max}-X_{min})$,CurrentSampleTime=0.

Fig.5 Algorithm of updating network size estimation

图 5 节点规模估计值更新算法

3.5 其他应用

利用 SEBSA 的哈希邻居维护协议,可以求解节点所关注属性值的加权和以及均值.由于 SEBSA 的哈希邻居表将整个覆盖网组织成一个 L 正则网,其中,L 是哈希邻居表的长度.假设节点 P_i 的节点标识为 X_i,对应的哈希邻居表为 NL_i,关注的属性值为 δ_i.扩展 SEBSA 的哈希邻居列表的条目信息,将属性值 δ 注入到哈希邻居表条目中,设节点 P_i 发起 δ 的求和过程,初始时,δ 的加和 $\Delta=0$.Δ 的计算过程如下:

(1) P_i 计算哈希邻居表中比 X_i 大的节点集合所对应的 δ 的加和,$\Delta=\Delta+\sum\{P_j\mid X_j>X_i,X_j\in NL_i\}$;

(2) 假设 P_{max} 为 P_i 哈希邻居表中最大节点标识所对应的节点,则 P_i 将 Δ 转发给 P_{max};

(3) P_{max} 重复步骤(1)、步骤(2),直到在 P_{max} 的邻居列表中再次找到 X_i 为止.

已知 δ 的加和 Δ 及节点规模估计值 N,则 δ 的均值为 Δ/N.由于所有节点标识服从(0,1)区间上的均匀分布,可知每个节点所计算的 δ 的个数期望为值 $L/2$.因此,求解 Δ 的跳步数为 $2N/L$.对于较大的 N,其计算延迟较大,可采取分段区间求和以减少计算延迟.由于每个节点的覆盖网邻居维护协议中维护了多个随机获取的节点标识,选择 m 个标识间距最大的节点标识构成向量 $V=(P_1,P_2,\ldots,P_m)$,同时发起求 Δ 的过程.当 P_j 找到 P_{j+1} 时算法停止,并将结果返回给发起节点 P_i,这样就可以进一步减少 Δ 的求解时间.

4 理论分析

SEBSA 的收敛性是我们关注的重点,这里的收敛性包含两部分内容:稳定时间,即节点的哈希邻居表收集到距离自身节点标识最近的一组节点所需要的时间;采样时间,即节点从稳定时间开始,直到将节点的规模估计值控制在特定误差范围内所需要的时间.本节首先引入基本概念和假设以便于数学分析,其次分析节点哈希邻居表区间长度的收敛速度以确定稳定时间,最后分析节点哈希邻居表区间长度的期望和方差以给出在特定误差范围内所需要的最小采样时间.

4.1 基本概念

假设网络中共有 N 个节点 $P_1,P_2,\ldots,P_N$,N 个独立同分布服从(0,1)区间上均匀分布的随机变量 $X_1,X_2,\ldots,X_N$ 分别为 $P_1,P_2,\ldots,P_N$ 的节点标识,$X_1,X_2,\ldots,X_N$ 由小到大的排序用 $X_{(1)},X_{(2)},\ldots,X_{(N)}$ 表示,即 $X_{(1)}\leqslant X_{(2)}\leqslant\ldots\leqslant X_{(N)}$.

$S_1,S_2,\ldots,S_N$ 分别为节点 $P_1,P_2,\ldots,P_N$ 的哈希邻居表,$S_i(1\leqslant i\leqslant N)$中共有 L 个元素,且按照从小到大顺序排列,即 $S_i=\{X_{i_1},X_{i_2},\ldots,X_{i_L}\mid X_{i_j}\leqslant X_{i_{j+1}},1\leqslant j\leqslant L\}$,$U(S_i)=X_{i_L}-X_{i_1}$,表示 S_i 的区间长度.初始时,$S_i(1\leqslant i\leqslant N)$从 N 个节点中随机均匀获取了 L 个节点.

$W_1,W_2,\ldots,W_N$ 分别为节点 $P_1,P_2,\ldots,P_N$ 的覆盖网邻居维护协议中的邻居列表,$W_i(1\leqslant i\leqslant N)$中共有 C 个元素.

4.2 稳定时间

稳定时间是指节点 P_i 的哈希邻居表 S_i 收集到距离 X_i 最近的 L 个节点所需要的时间,此时,称 S_i 达到稳定状态.假设在 t 时刻 S_i 的区间长度的期望为 $E(U(S_i^t))$,且 S_i 达到稳定时的区间长度期望为 $E(U(S_i))$,则可以分析 $E(U(S_i^t))$ 与 $E(U(S_i))$的差错程度以确定 S_i 的收敛时间.

定理 1. S_i 稳定时的区间长度期望 $E(U(S_i))$为 $\dfrac{L-1}{N+1}$.

证明:由均匀分布顺序统计量的性质可知 $E(X_{(i)})=\dfrac{i}{N+1}$,$E(U(S_i))$等价于求解包含 X_i 在内的 L+1 个连续随机变量的区间长度的期望.假设 X_{i_0} 对应 $X_k(1\leqslant k\leqslant N)$,则

$$E(U(S_i))=E(X_{i_L})-E(X_{i_0})=\frac{L-1+k}{N+1}-\frac{k}{N+1}=\frac{L-1}{N+1}. \qquad \square$$

在 SEBSA 中,每个节点选择自己哈希邻居表中的一个节点以相互交换各自的哈希邻居表.假设 t 时刻 $X_j\in S_i^t(1\leqslant j\leqslant N,1\leqslant i\leqslant N)$,且节点 P_i 选择了 P_j 相互交换哈希邻居表.理想情况下,$X_{i_1}\cong X_{j_1}$且$X_{i_L}\cong X_{j_L}$.此时,在区间$[X_{i_1},X_{i_L}]$上共有$2L-D_{ij}^t\,|S_i^t|$个点,其中,D_{ij}^t 为 t 时刻 S_i 和 S_j 中重复点个数占 S_i 节点个数的比例,即 $D_{ij}^t=|S_i^t\cap S_j^t|/|S_i^t|$.在区间$[X_{i_1},X_{i_L}]$上的点满足均匀分布,则

$$E(U(S_i^{t+1}))=E(U(S_i^t))/(2-D_{ij}^t) \tag{1}$$

显然,D_{ij}^t 与 $U(S_i^t)$ 成反比,即

$$D_{ij}^t\propto\frac{1}{E(U(S_i^t))} \tag{2}$$

且当 $E(U(S_i^t))=\dfrac{L-1}{N+1}$时 $D_{ij}^t=1$,可知 $D_{ij}^t=\dfrac{L-1}{(N+1)E(U(S_i^t))}$.综合公式(1)、公式(2)可得:

$$E(U(S_i^{t+1}))=E(U(S_i^t))\Big/\left(2-\frac{L-1}{(N+1)E(U(S_i^t))}\right) \tag{3}$$

假设 $E(U(S_i^0))=1,L=40,N=10000$,则 t=14 时可使 $E(U(S_i^t))$ 收敛于 $\dfrac{L-1}{N+1}$.由于节点交换哈希邻居表可以使每个节点在每一轮中的区间长度平均收缩两次,即大约经过 7 轮即可保证节点的区间长度稳定在 $\dfrac{L-1}{N+1}$.

4.3 采样时间

采样时间是指从节点 $P_i(1\leqslant i\leqslant N)$的哈希邻居表 S_i 达到稳定时间开始,直到节点的规模估计值与真实值的误差不大于δ所需要的时间,节点 P_i 通过覆盖网邻居维护协议采样其他节点的规模估计值以减少估计误差.由定理 1 可知 $U(S_i)$的期望,若获得各个 $U(S_i)$方差的上界,则利用中心极限定理可以求解出需要在特定采样次数下节点规模估计值的相对差错程度.

定理 2. S_i 达到稳定状态时的区间长度方差 $D(U(S_i))$不大于 $\left(\dfrac{(L-1)\log N}{N}\right)^2-\left(\dfrac{L-1}{N+1}\right)^2$.

证明:$X_{(1)},X_{(2)},\ldots,X_{(N)}$是(0,1)区间上的均匀分布顺序统计量,设$\Delta_N$代表 $X_{(1)},X_{(2)},\ldots,X_{(N)}$中连续两点间的最大距离,$\Delta_N=\max\limits_{0\leqslant u<v\leqslant 1}\{v-u:X_{(i)}\notin(u,v),1\leqslant i\leqslant N\}$.

Ralph 等人[15]证明:当 $N\to\infty$时,$\limsup\limits_{x\to\infty}\Delta_N=1-\exp\left(-\dfrac{\log N}{N}\right)\cong\dfrac{\log N}{N}$.故

$$\limsup_{x\to\infty}U(S_i)\leqslant\frac{(L-1)\log N}{N} \tag{4}$$

因此有

$$D(U(S_i))=E(U^2(S_i))-[E(U(S_i))]^2=\frac{1}{N}\sum_{i=1}^{N}U^2(S_i)-[E(U(S_i))]^2\leqslant\left(\frac{(L-1)\log N}{N}\right)^2-\left(\frac{L-1}{N+1}\right)^2 \quad (5)$$

□

假设节点 P_i 独立采样了 n 组哈希邻居表的区间长度 $U(S_{i_1}),U(S_{i_2}),...,U(S_{i_n})$,由中心极限定理可知:

$$Y=\frac{\sum_{j=1}^{n}U(S_{i_j})-nE(U(S_i))}{\sqrt{D(U(S_i))}\sqrt{n}}\sim N(0,1) \quad (6)$$

设 δ 为 $\sum_{j=1}^{n}U(S_{i_j})$ 相对于 $nE(U(S_i))$ 的误差程度,由定理 1 和定理 2 的结论可得:

$$P\left(\left|\sum_{j=1}^{n}U(S_{i_j})-nE(U(S_i))\right|<\delta nE(U(S_i))\right)=P\left(|Y|<\frac{\delta nE(U(S_i))}{\sqrt{D(U(S_i))}\sqrt{n}}\right)\cong P\left(-\frac{\delta\sqrt{n}}{\log N}<Y<\frac{\delta\sqrt{n}}{\log N}\right)=2\Phi\left(\frac{\delta\sqrt{n}}{\log N}\right)-1 \quad (7)$$

当 $2\Phi\left(\frac{\delta\sqrt{n}}{\log N}\right)-1\geqslant 0.95$ 时,查表可得 $\frac{\delta\sqrt{n}}{\log N}\geqslant 1.96$.即 $n\geqslant\left(\frac{1.96\log N}{\delta}\right)^2$ 时,可保证 95%的节点的规模估计值与真实节点规模的差错率不超过 δ.

而在 SEBSA 中,CYCLON 方法的邻居列表大小为 C,每次交换的邻居数目为 $C/2$.理想情况下,节点 P_i 在 t 时刻采样的邻居列表的数目为 $(C/2)^t$.当 δ=0.05 时,$t\geqslant 2\log_{C/2}(39.2\log N)$即可.

令 δ=0.05,C=20,N=10000,则 $t\geqslant 6$ 时,即节点规模估计值的误差范围在 5%以内的概率不小于 95%.

5 性能测试

SEBSA 的目标是在动态波动的大规模网络环境下,实现较高的规模估计精度、鲁棒性以及较小的计算延迟,并有效权衡通信开销和存储开销.反熵聚集方法(anti-entropy aggregation protocol,简称 AAP)[5]通过迭代计算反熵信息,以达到高计算精度和快速收敛;抽样碰撞方法(sample and collide,简称 SC)[9]采用并行的多个随机行走过程和设置随机行走路径上重复节点数目等方式以提高计算精度和鲁棒性,且通信开销相对较小;另外,区间密度方法(interval density approach,简称 IDA)[10]收集特定区间上的节点数目,具有简单鲁棒的优点.因此,为了检验 SEBSA 在计算精度、鲁棒性、计算延迟以及系统开销的效果,我们基于 PeerSim[16]模拟器实现了 SEBSA,AAP, SC 以及 IDA,实验内容包括:

(1) SEBSA 收敛性.指 SEBSA 哈希邻居表的稳定时间和采样时间;

(2) 计算精度.即在动态网络环境下,节点的规模估计值相对于真实网络大小的差错程度;

(3) 鲁棒性.即在节点发生失效、丢包情况下,节点的规模估计值相对于真实值的差错程度;

(4) 计算延迟.即每次计算过程的收敛时间;

(5) 系统开销.包括节点在计算过程中的通信开销和存储开销.

由于不同节点上计算出的规模估计值各不相同,我们采用平均相对差错率(mean relative error,简称 MRE)以评估节点的规模估计值与真实网络大小的差错程度.具体来讲 $MRE=\frac{1}{N}\sum_{i=1}^{N}\left|\frac{\varphi_i-N}{N}\right|$,其中,$\varphi_i$ 为节点 i 的规模估计值,N 为真实的节点规模大小.

为了保证节点连通性,在 PeerSim 中实现了 CYCLON 协议以周期性地维护节点邻居列表.其中,邻居列表的大小为 20,每一轮中每个节点从邻居列表中抽取 10 个节点相互交换.实验中,所有方法的邻居维护都基于该 CYCLON 协议.

在 SEBSA 中,节点通过与哈希邻居表中的节点相互交换最近邻以达到快速提高计算精度的目标,并通过计算相邻两个节点之间的平均距离以确定节点规模大小.因此,其哈希邻居表大小 L 对计算精度影响较大.图 6 显示了 SEBSA 在拥有不同哈希邻居表时,平均相对差错率随时间的收敛速度.从中可以看出,L 越大,则计算精度越高,收敛速度越快.实验中,设置 L=40 即可保证在 40 轮时具有小于 3%的平均相对差错率.

AAP 中假设了所有节点能够同步地以一定概率作为反熵聚集的发起节点,这在实际网络中是不现实的,因为每个节点的时钟并不能达到完全同步.在实验中,我们认为该假设是成立的,即实现了理想情况下的 AAP.在 AAP 中,每次反熵聚集过程需要一定的计算轮数以保证计算精度,以 40 轮为一个周期以保证 GBA 的收敛性.

SC 采用随机行走的方式估计节点规模,而在动态网络中,某个节点的丢失会造成其随机行走消息的丢失,因此对动态环境的适应能力有限.为了便于对比,我们对 SC 采用消息确认机制,即随机行走消息每到达一个节点,就向其前继节点发送确认消息.如果节点在一定时间间隔内未收到确认消息,则重启该随机行走过程.SC 中,重复出现的节点次数 L 越大,其计算精度越高,设置 L=50.

IDA 方法则周期性地从 CYCLON 的邻居列表中筛选落在关注区间上的节点即可.另外,采用 Gossip-Style Failure Detection[17]的方法检测节点失效,该失效检测时间为 $O(IN\log N)$.其中,I 为关注空间的长度,N 为节点规模.实验中设置 $I=\sqrt{N}/N$,则可保证节点规模估计的差错率在 $\sqrt{\log N}/N^{1/4}$ 范围内.

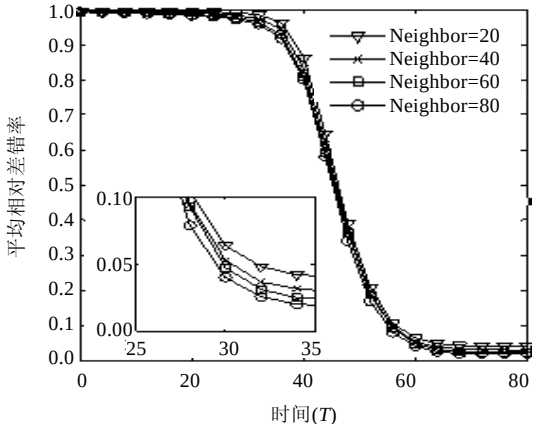


Fig.6 Convergence with different HNL

图 6 不同哈希邻居表下的收敛性

5.1 SEBSA收敛性

SEBSA 的收敛性包括两部分:一是哈希邻居表的稳定时间,即哈希邻居表收集到距离自身节点标识最近的一组节点所需要的时间;二是哈希邻居表的采样时间,即节点从稳定时间开始,直到将节点的规模估计值控制在特定误差范围内所需要的时间.

图 7 显示,当 N=10000 时,理论分析的哈希邻居表稳定时间与实际实验中哈希邻居表稳定时间的对比效果,其中,纵轴代表节点哈希邻居表的平均区间长度.可以看出,实际实验中的哈希邻居表稳定时间与理论分析的结果相同,即在 T=6 时哈希邻居表达到稳定状态;另外,在 T=1 时刻,实际实验中哈希邻居表区间收敛的速度好于理论结果;而在 T>1 以后的时刻,理论解析结果则稍好于实验效果.由于在 T=0 时,在 SEBSA 中各个节点是通过随机行走寻找该路径上距离自己最近的节点以初始化哈希邻居表,因此,该哈希邻居表的节点在哈希空间内并不满足(0,1)区间上的均匀分布.这样,在 T=1 时,各个节点就可以以更快的速度收敛;而在 T>1 以后的时刻,也是由于各个节点哈希邻居表中节点标识的分布在(0,1)区间上不满足均匀分布,这就使得哈希邻居表交换过程中重复节点的比例会比理论分析中的更高,因此其收敛速度相对稍慢.

图 8 显示,当各个节点的哈希邻居表稳定后,各个节点在采样过程中其规模估计值差错率随时间的累积分布情况.可以看出:在 T=20 时,有 92.5%的节点规模估计值的差错率小于 6%,有 96.2%的节点规模估计值的差错率小于 7%;而理论分析中,T>5 时节点规模估计值的差错率小于 5%的概率不小于 95%.由于在实际实验中各个节点采样的结果中会有大量的重复采样,因此不可能达到理论上完全没有重复采样的指数级增长效果.

综上可以看出,SEBSA 的哈希邻居表在 T=6 时即可达到稳定状态;而哈希邻居表稳定后,再经过 20 轮次的时间即可保证有 92.5%的节点规模估计值的差错率小于 6%.因此,SEBSA 表现出了较高的收敛性.

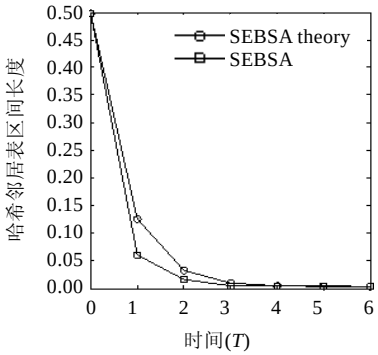


Fig.7 Stable time

图 7 稳定时间

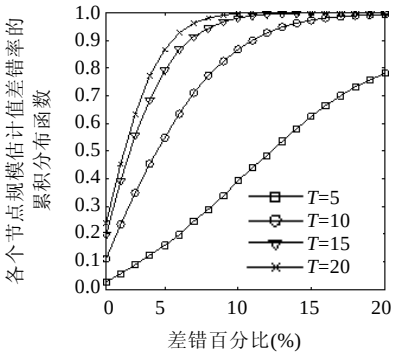


Fig.8 Sample time

图 8 采样时间

5.2 计算精度

为了验证不同网络模型下方法的计算精度,设置网络的初始规模为 10 000,设计实现了两种动态网络模型:第 1 种是节点波动网络,节点规模在[9000,11000]之间周期性波动.即在每一轮中,节点数目要么增加要么减少,当节点数目增长至 11 000 时,则改为节点减少,当节点数目减少至 9 000 时,则改为节点增长,每一轮变化的比例为 0.1%;第 2 种是节点替换网络,即每一轮中同时有一定比例的节点加入和退出,并保持加入和退出的节点数相同,每一轮节点加入和退出的比例各为 0.1%.

图 9 显示了在节点周期性波动网络中各个协议随时间的计算精度对比.可以看出:SEBSA 的平均相对差错率稳定在 6%以下;AAP 在大部分时间上平均相对差错率都较低,但是当 T=720 和 T=760 时,相对差错率突然变大,分别达到了 13%和 47%;SC 的平均相对差错率在 1%~40%的较大范围内波动;IDA 的平均相对差错率整体偏差较大,且出现大幅的周期性波动现象.在波动网络中,SEBSA 在节点增长阶段对于刚刚加入的节点还没有完全收敛,这在一定程度上使得整体的平均差错率稍高;而节点减少阶段,每个节点周期性和哈希邻居表中的其他节点交换各自探测出的失效节点信息,因此可以快速发现失效节点,其整体的平均差错率较低.AAP 在反熵聚集计算前期,若被去除的节点上携带有较大的规模估计信息,则会造成整体的估计值偏高.例如,当 T=720 和 T=760 时, AAP 就出现了较大的估计偏差.在波动网络环境下,SC 在随机行走收集到的节点信息很可能已有一部分失效,这些失效节点一定不会再出现在网络中,这样就可能引起较大的差错;IDA 中仅从覆盖网邻居维护协议的邻居列表上收集节点信息,当节点规模增长时,其发现新节点的能力较弱;当节点规模减少时,其失效节点检测时间过长.也就是说,IDA 的平均差错率周期性变化的特点是由节点规模的周期性变化导致的.

图 10 显示了在节点周期性替换网络中计算精度的对比情况.

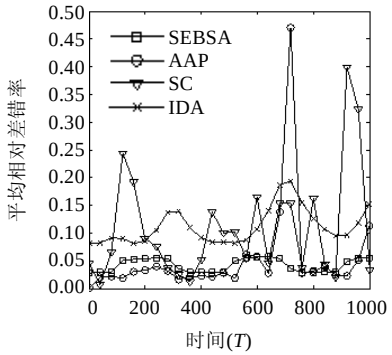


Fig.9 Mean relative error in fluctuated network

图 9 节点波动网络中各协议的计算精度

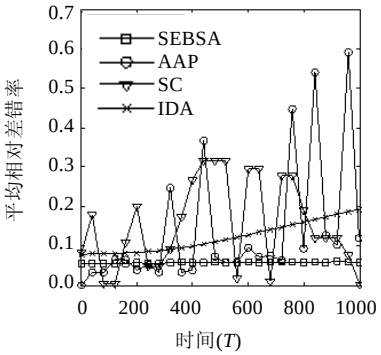


Fig.10 Mean relative error in substituted network

图 10 节点替换网络中各协议的计算精度

由图 10 可以看出:SEBSA 的计算精度几乎稳定在 6%以下;AAP 出现了多次的差错率较大的情况;SC 表现出较强的不稳定性,且平均相对差错率的波动范围较大;IDA 的平均相对差错率随时间递增.SEBSA 周期性地与哈希邻居表中其他节点交换信息,一方面可以使节点的哈希邻居表快速收敛,另一方面也可以有效地检测失效节点,因此对动态网络的适应能力较好;在 AAP 中,一部分节点周期性被替换,使得携带有较大规模估计信息的节点失效的概率更高,因此其出现偏差的次数也越多;在节点频繁加入退出情况下,SC 随机行走的长度具有较大的不确定性,因此其精度较差;IDA 无法使新加入的节点快速收敛,且失效检测的时间过长,造成在不断有节点加入时其差错程度逐渐放大.

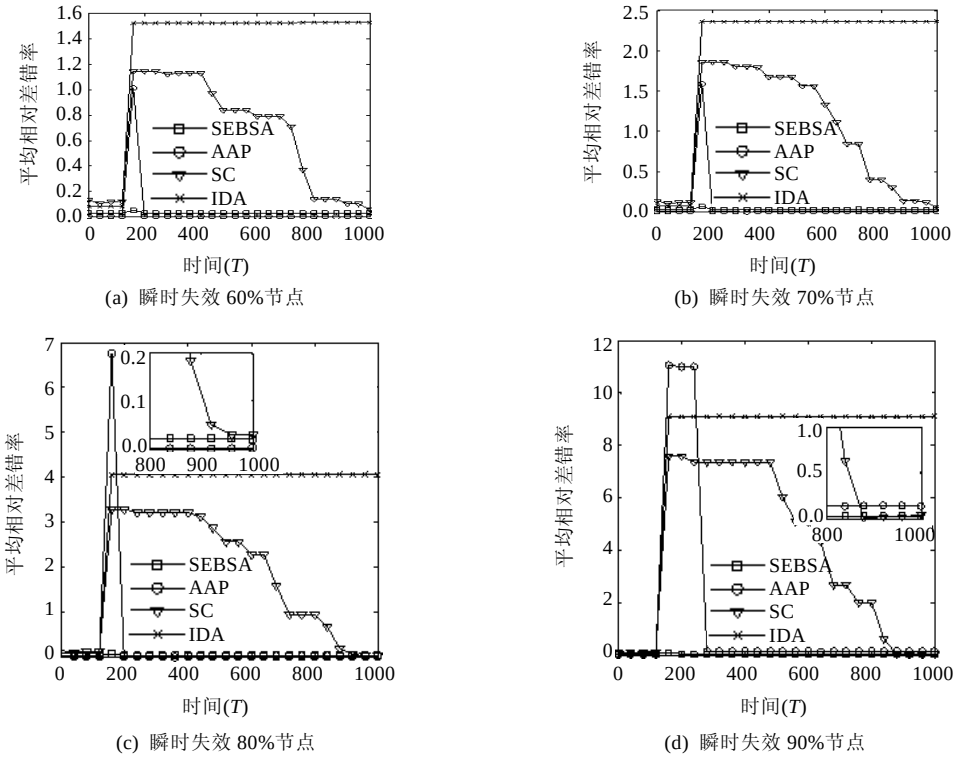
综上两种动态网络模型可以看出,相对于 AAP,SC 和 IDA,SEBSA 具有更好的计算精度和稳定性,在节点频繁加入退出时可以保证各个节点快速收敛到较高的计算精度.

5.3 鲁棒性

对于节点规模估计方法,节点失效和数据链路的可靠性会对其性能造成影响.因此,我们通过检测节点失效和丢包率对节点规模估计方法的影响程度来衡量系统的鲁棒性.具体来说,分两类情况对 SEBSA,AAP,SC 以及 INA 的鲁棒性进行对比:大量节点瞬间失效;每一轮数据包在传输时有一定的丢包率.

5.3.1 瞬时失效

为了验证大规模节点失效后方法的自适应能力,实验中初始节点规模为 10 000,在 T=165 时发生失效,失效后节点规模保持不变,分别测试了系统中 60%,70%,80%以及 90%的节点失效后的情况(如图 11 所示).



(a) 瞬时失效 60%节点 (b) 瞬时失效 70%节点

(c) 瞬时失效 80%节点 (d) 瞬时失效 90%节点

Fig.11 Mean relative error with instantaneous node failure

图 11 瞬时失效时平均相对差错率

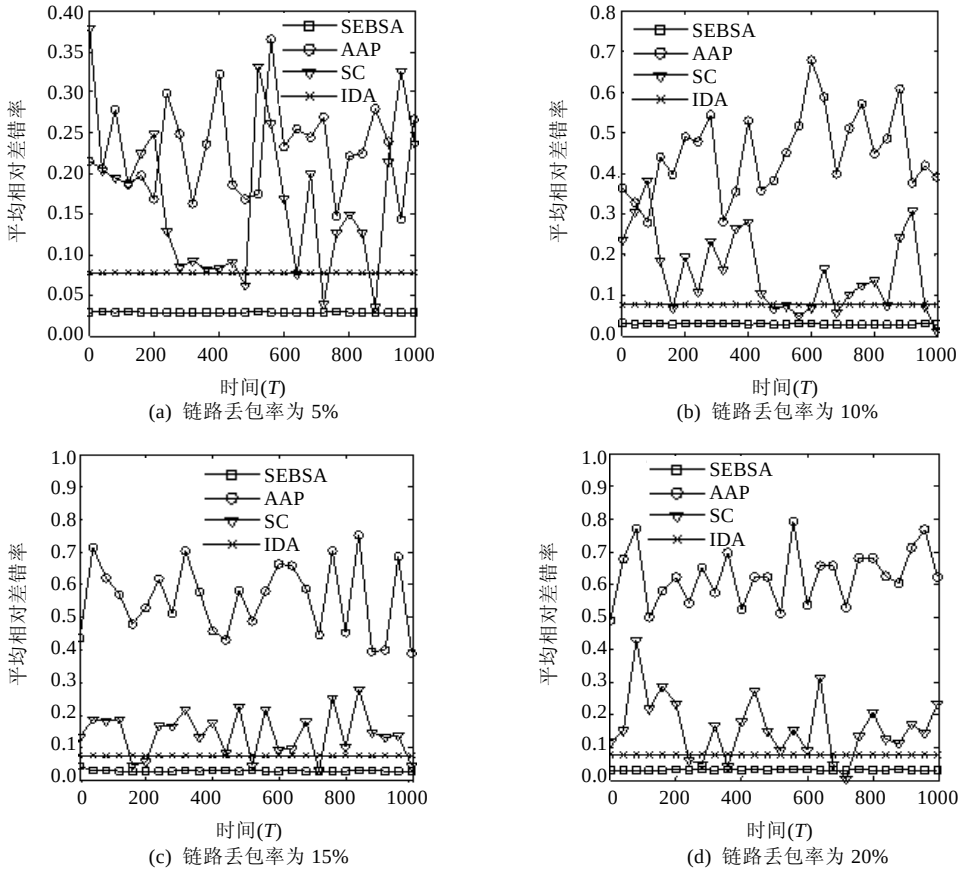
从图 11 中可以看出:SEBSA 在失效瞬间的周期内,其规模估计值就能恢复到较高的精度,在失效的下一个周期即可恢复到稳定状态;AAP 也可以在失效后的一个周期内达到稳定状态,且在 60%,70%,80%的节点失效恢复后的精度与失效前保证不变,90%节点失效恢复后其平均相对差错率有略微增加;SC 需要较长的时间才能达

到将差错程度控制在可接受的范围;IDA 很难在短时间内检测到失效节点,其平均相对差错率较大.SEBSA 的各个节点在与哈希邻居表中的节点交换失效节点过滤器的过程中,可以快速检测到失效节点;AAP 周期性重启保证了节点失效后的恢复能力,另外,在 90%节点失效时,AAP 产生的计算差错是由 CYCLON 协议不能保证覆盖网的全连通性造成的;SC 随机行走过程本身的计算时间较长决定了其失效后的收敛时间也较大;IDA 在有限邻居列表情况下,采用基于 Gossip 的失效发现服务的失效检测时间过长,造成了长时间内较大的差错程度.

综上可以看出,相对于 AAP,SC 和 IDA,SEBSA 在节点大规模失效后具有快速恢复能力,且失效后仍然保持了较高的计算精度.

5.3.2 丢 包

图 12 显示了节点规模为 10 000 的静态网络中链路丢包率为 5%,10%,15%,20%时各方法计算精度的对比情况.由于各种方法都采用 CYCLON 方法实现覆盖网邻居维护,因此这里不考虑覆盖网邻居维护所造成的链路丢包,而只考虑在失效节点检测、维护上层数据结构所造成的丢包.



(a) 链路丢包率为 5% (b) 链路丢包率为 10%

(c) 链路丢包率为 15% (d) 链路丢包率为 20%

Fig.12 Mean relative error with drop package

图 12 链路丢包时平均相对差错率

可以看出:SEBSA 几乎不受链路丢包的影响,其平均相对差错率稳定在 3%左右;AAP 的平均相对差错率最大,且随着丢包率的增加而增加;SC 的平均差错率较大且不稳定,但相对于 AAP 稍好;IDA 方法几乎没有受到链路丢包的影响.在 SEBSA 中,节点通过采样邻居节点信息以估计节点规模,当丢包发生时,某个节点没有及时收到最近邻更新消息,这并不影响节点的估计值;在 AAP 方法反熵聚集计算中,如果聚集交换的两个节点中只有一个节点获得了对方的估计值,而另一个节点由于丢包没有收到对方的估计值,就会使节点的加权和不为 1,从

而引起计算错误,且该错误的概率随着丢包率的增大而增大;SC 方法中,随机行走的下一条不确定性使得其平均相对差错率的波动范围较大;IDA 中仅仅从下层收集节点标识,而并没有额外的通信开销,因此链路丢包不会对其造成影响.可见,相对于 AAP 和 SC,链路丢包对 SEBSA 和 IDA 具有较低的影响程度,且 SEBSA 保持了较高的计算精度.

5.4 计算延迟

图 13 显示了各种方法在 10 000 个节点规模下从开始计算到结果收敛的计算延迟.可以看出:SEBSA 和 AAP 在 30 轮的时间内即可达到较高的计算精度,且在计算前期 SEBSA 的收敛速度较 AAP 更快;SC 和 IDA 则要经过相当长的计算时间才能收敛到较精确的结果.其中,SC 的收敛过程不稳定,而 IDA 的收敛时间最长.

SEBSA 基于哈希值吸引语义上临近的节点能够保证哈希邻居表的快速收敛性,而指数级迭代采样邻居节点的规模估计值能够保证节点规模估计值在短时间内达到较高精度;AAP 的反熵聚集计算方式保证了节点规模估计值能够指数级收敛;SC 随机行走过程占用了较长的计算时间,且其收敛过程的不稳定性是由随机行走过程碰撞到下一个已重复节点的时间的不确定性决定的;而 IDA 只在有限的邻居列表中采样节点决定了其计算延迟较长.

可以看出,SEBSA 在静态网络环境下表现出了与 AAP 相当的收敛能力,保证了节点规模估计值在较短的时间内达到较高的计算精度.

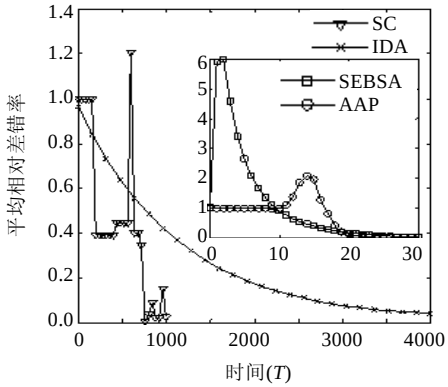


Fig.13 Computing delay

图 13 计算延迟

5.5 系统开销

5.5.1 通信开销

在计算通信开销时,由于各种方法基于 CYCLON 实现覆盖网邻居维护,因此可假定 CYCLON 在各种方法中具有相同的通信开销.IDA 方法从节点采样协议中获取其他节点的标识信息,并没有引入额外的通信开销.图 14 显示了 SEBSA,AAP 和 SC 所有节点总的通信开销.可以看出:SEBSA 与 AAP 相比,具有相对较小的通信开销;SC 的通讯开销相对于 SEBSA 和 AAP 最小;SEBSA 与 AAP 相比,每个周期内各个节点都会主动发起通信,而 AAP 中同一个周期内可能有多个节点同时发起反熵聚集计算,这就带来了更多通信开销;而 SC 尽管每个周期都会发起新的随机行走过程,但每个随机行走路径的长度有限,因此仍然具有较低的通信开销,且增长缓慢.

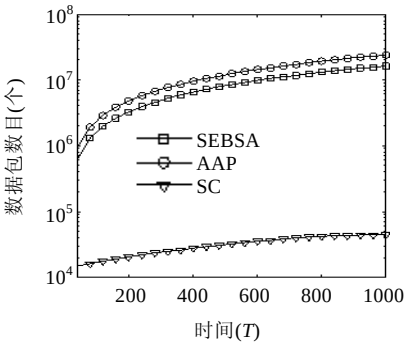


Fig.14 Transmission overhead

图 14 通信开销

5.5.2 存储开销

假设每一个节点标识占用 16 字节的存储空间,节点规模为 N.在 SEBSA 中,每个节点的存储开销为哈希邻居表的存储开销大小,每个表项包括节点标识 16 字节,节点 IP 占用 4 字节以及端口号占用 2 个字节,设哈希邻居表长度为 C,因此其存储开销为 $22C$;而 AAP 的存储开销最小,即每个节点只保存一个节点规模估计值,其存储开销为一个 4 字节的整数;在 SC 中,采用超时重启的方式以保证随机行走过程的连续性,对于每个随机行走过程,其路径长度为 $O(\sqrt{LN})$,其中,L 为随机行走中重复节点出现的次数.因此,每个节点为了存储该随机行走消息需要保存该随机行走路径上的所有节点信息,包括节点标识、IP 地址及其端口,其存储开销至少为 $22\sqrt{LN}$;而在 IDA 方法中,假设关注的空间为 I,则在空间 I 上的节点标识的期望个数为 IN,因此其存储开销为 $22IN$.

表 1 显示了节点规模为 10 000、在 40 轮的时间内进行一次规模估计时各种方法的通信开销和计算开销的对比情况.可以看出,SEBSA 实现了通信开销和存储开销的有效折中.

Table 1 Overhead of 40 rounds

表 1 40 轮时间内的开销

算法	SEBSA (C=40)	AAP	SC (L=50)	IDA (I=0.01)
通信开销(个)	644K	972K	4K	0
存储开销(字节)	880	4	15.6K	2.2K

6 结束语

本文提出一种基于语义吸引的节点规模估计方法 SEBSA.SEBSA 通过采样哈希邻居表中节点哈希值的平均间距,以保证节点规模估计精度;哈希值临近的节点周期性交换哈希邻居表以吸引更临近的节点,从而使哈希邻居表快速收敛.通过采样多组节点的规模估计值保证了计算精度,周期性交换哈希空间上的邻居信息保证了在动态环境的鲁棒性.理论分析和实验结果表明,相对于已有方法,SEBSA 在节点频繁波动、大规模失效、链路丢包等情况下仍然具有较高的计算精度和鲁棒性.另外,SEBSA 还在通信开销和存储开销之间实现了较高的折中.下一步值得研究的问题是考虑 SEBSA 中哈希邻居表大小对规模估计值的影响程度.该问题有助于设计不同的哈希邻居表大小以适应不同规模估计精度的需求,从而在规模估计精度和存储开销之间更好的折中.

References:

[1] Rowstron A, Druschel P. Pastry: Scalable, decentralied object location and routing for large-scale peer-to-peer systems. In: Proc. of the IFIP/ACM Int'l Conf. on Distributed Systems Platforms. 2001. 329−350. [doi: 10.1007/3-540-45518-3_18]

[2] Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for Internet applications. In: Proc. of the 2001 SIGCOMM Conf. New York: ACM Press, 2001,31(4):149−160. [doi: 10.1145/383059.383071]

[3] Ganesh AJ, Kermarrec AM, Massoulié L. Peer-to-Peer membership management for gossip-based protocols. IEEE Trans. on Computers, 2003,52(2):139−149. [doi: 10.1109/TC.2003.1176982]

[4] Baldoni R, Beraldi R, Quema V, Querzoni L, T-Piergiovanni S. TERA: Topic-Based event routing for peer-to-peer architectures. In: Proc. of the 2007 Inaugural Int'l Conf. on Distributed Event-Based Systems. New York: ACM Press, 2007. [doi: 10.1145/1266894.1266898]

[5] Jelasity M, Montresor A. Epidemic-Style proactive aggregation in large overlay networks. In: Proc. of the Int'l Conf. on Distributed Computing Systems. 2004. 102−109. [doi: 10.1109/ICDCS.2004.1281573]

[6] Bawa M, G-Molina H, Gionis A, Motwani R. Estimating aggregates on a peer-to-peer network. Technical Report, Technical Department of Computer Science, Stanford University, 2003.

[7] Flajolet P, Martin GN. Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences, 1985,31(2):182−209. [doi: 10.1016/0022-0000(85)90041-8]

[8] Kennedy O, Koch C, Demers A. Dynamic approaches to in-network aggregation. In: Proc. of the IEEE 25th Int'l Conf. on Data Engineering. 2009. 1331−1334. [doi: 10.1109/ICDE.2009.233]

[9] Massoulié L, Le Merrer E, Kermarrec AM, Ganesh A. Peer counting and sampling in overlay networks: Random walk methods. In: Proc. of the 25th Annual ACM Symp. on Principles of Distributed Computing. ACM Press, 2006. 123−132. [doi: 10.1145/1146381.1146402]

[10] Kostoulas D, Psaltoulis D, Gupta I, Birman K, Demers A. Decentralized schemes for size estimation in large and dynamic groups. In: Proc. of the 4th IEEE Int'l Symp. on Network Computing and Applications. 2005. 41−48. [doi: 10.1109/NCA.2005.15]

[11] Cardoso JCS, Baquero C, Almeida PS. Probabilistic estimation of network size and diameter. In: Proc. of the 4th Latin- American Symp. on Dependable Computing. 2009. 33−40. [doi: 10.1109/LADC.2009.19]

[12] Shafaat TM, Ghodsi A, Haridi S. A practical approach to network size estimation for structured overlays. In: Proc. of the IWSOS 2008. 2008. 71−83. [doi: 10.1007/978-3-540-92157-8_7]

[13] Voulgaris S, Gavidia D, Van Steen M. CYCLON: Inexpensive membership management for unstructured P2P overlays. Journal of Network and Systems Management, 2005,13(2):197−217. [doi: 10.1007/s10922-005-4441-x]

[14] Russo RP, Rothmann MD. Maximal divergent uniform spacings. Stochastic Processes and their Applications, 1989,33(1): 175−183. [doi: 10.1016/0304-4149(89)90074-4]

[15] Peersim. http://peersim.sourceforge.net

[16] Van Renesse RV, Minsky Y, Hayden M. A gossip-style failure detection service. In: Proc. of the Middleware'98. 1998.

马行空(1987－),男,河南邓州人,博士生,CCF 会员,主要研究领域为网络计算,数据分发技术.

王意洁(1971－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为网络计算,海量数据处理,并行与分布处理.

郑重(1982－),男,博士生,CCF 会员,主要研究领域为网络计算,数据分发技术.