应用层组播

背景技术

（IP组播与应用层组播）

1. 选举簇首

所有节点自组织成mesh网，并将所有节点按照区域进行分簇。本文提出的方案的应用场景为战术通信网络，战术通信网络的节点隶属于某个团、连、排、班。因此，节点在分簇之后，则固定属于某个簇。若节点移动至其它簇的区域范围，假设节点不会长时间停留在其它的簇，过一段时间，节点会移动至其归属的区域。将节点分簇后，为提高簇与簇之间的通信效率，需要为每个簇选举一个簇首，用于维护簇与簇之间信息同步。

文献[2]选举簇首及辅助簇首的方法，仅考虑了簇内节点间的距离，或者是仅考虑了节点的出度及传输时延，而没有考虑节点的移动性，应用层组播节点的移动性大，如果没有考虑组播节点的移动性，推举出的簇首可能是非常不稳定的节点。文献[3]推举簇首的方法则仅考虑了节点的移动性，没有考虑节点的出度和传输时延，这种情况下选举出的簇首的转发能力和传输效率可能会满足不了需求。本文选择簇首的方法，充分考虑了节点的移动性、数据转发能力和传输时延等因素，在节点移动的情况下，保证数据传输的可靠性和高效性。

* 1. 节点稳定度

首先说明两个参数：链路过期时间（LET）和节点稳定度。

参考文献[1]，根据链路中两个节点的位置、移动方向和移动速度，可以预测该链路的过期时间LET，其定义为：



其中：



（,）、（,）分别为节点的二维空间坐标; 、分别为节点的移动速度值; 、分别为节点的移动方向。

根据LET，定义节点的稳定度为该节点与其所有相邻节点组成的链路的LET之和，即：



其中为节点的相邻节点集合。

* 1. 节点优先值

节点的优先值，考虑的因素有节点的稳定度、节点的出度、节点的最大出度和节点的传输时延。其定义如下：



其中V为拓扑图的节点集合，是[0,1]之间的平衡系数，用于协调稳定度、节点度数和节点传输时延的权重。、分别为节点及其邻节点的出度之和、拓扑中所有节点的出度之和，为以节点为源节点，其余节点为目的节点的传输路径的传输时延之和，。即：



若两个节点间有多条传输路径，取传输路径时延的平均值。

节点的稳定度越高，节点退出的概率就越小；节点及其相邻节点的度越大，则通过该节点接收到数据的节点越多，数据传播得越快，节点的转发能力就越好。节点的传输时延越小，则组播树的传播效率也越高，工作性能越好。因此，节点稳定性越高、节点及其邻节点的度越大、传输时延越低，节点的稳定性和数据转发效率就越高。可以看出，节点的优先值越低，节点的性能就越好。

* 1. 选举簇首的过程

1. 节点维护一个信息表info-list，存储节点的ip地址、稳定值、出度和传输时延。
2. 各节点向相邻节点发送探测报文，探测报文中包含信息表info-list的信息。随机选择某一节点作为暂定簇首，暂定簇首创建成员信息表member-list。当节点收到探测报文时，如果该节点不是暂定簇首，则转发该探测报文给其相邻节点；如果该节点是暂定簇首，并且暂定簇首的member-list中没有包含该探测报文的信息，将该信息添加至member-list中。
3. 经过一段时间，暂定簇首的成员信息表member-list中包含了所有节点的信息，并计算出所有节点的优先值。将优先值最小的节点命为簇首，次小值命为备用簇首。并向簇内所有成员发送选举出的簇首和备用簇首的信息。成员接收信息后，将簇首和备用簇首的信息记录在info-list中。同时，还需要处理member-list：
   1. 若暂定簇首不是选举出的簇首和备用簇首，则将member-list发送给簇首和备用簇首。成员信息表被接收并保存后，暂定簇首将member-list删除。
   2. 若暂定簇首是选举出的簇首，则暂定簇首只需记录备用簇首信息，并将member-list发送给备用簇首。
   3. 若暂定簇首是选举出的备用簇首，则暂定簇首只需记录簇首的信息，并将member-list发送给簇首。
4. 簇内传输路径构建

首先给出两个概念，若某条链路的两个节点属于不同的簇，则这两个节点称为中间节点(mediator)，这条链路称为中间链路(meLink)。中间节点需要保存中间信息（meInfo），包括节点连接到的簇的编号，节点连接到该簇的节点的IP地址。

簇内信息的传播采用广播的方式。以节点间的距离作为权值，并基于Dijkstra算法，构建簇内信息传输路径。通过簇内信息传输路径：（1）簇首可以将信息广播至该簇的所有成员；（2）簇内成员可以将信息发送给簇首，然后簇首再将信息广播至簇内的其它成员，或者是通过组播树，将信息发送至其它簇的簇首，其它簇的簇首再将信息广播至其成员，从而实现节点之间信息的互通。

若簇p中的某个节点i与相邻簇q的某个节点j之间存在中间链路，此时有两种情况：

（1）节点i与其簇首之间可达。则簇p的簇首不会通过簇内的信息传输路径将信息发送至节点j。节点j接收节点i的信息的方式为：首先，所有簇的簇首需要构建组播树，通过组播树，各个簇可以互通信息。簇q的簇首接收到了簇p的簇首发送的信息后，再通过其簇内信息传输路径，将信息发送至节点j。

（2）节点i与其簇首之间不可达。因为节点i与其簇首之间不可达，因此节点i不能通过其簇首获取其它簇的节点的信息。为了使节点i能获取其它簇节点的信息，节点j在接收到簇q的簇首发送的信息时，也应将此信息转发给节点i。若节点i与其所属簇内的部分节点之间是可达的，显示这部分节点与簇p的簇首也是不可达的，则节点i在收到由节点j转发的信息时，应该将信息也转发给这些节点。若节点i与簇q之间存在多条中间链路，则节点i可能会收到簇q多个节点转发的多份相同信息，此时节点i会只保留其中的一份信息，将其它的重复信息丢弃。

如图2.1所示，被圈出的部分即是簇3中与簇首不可达的节点。簇3的节点4与簇1的节点2之间存在链路，则节点2（簇1）在收到其簇首发送的信息时，应转发该信息给节点4（簇3），然后节点4再将信息转发给同一簇的节点2和节点3。

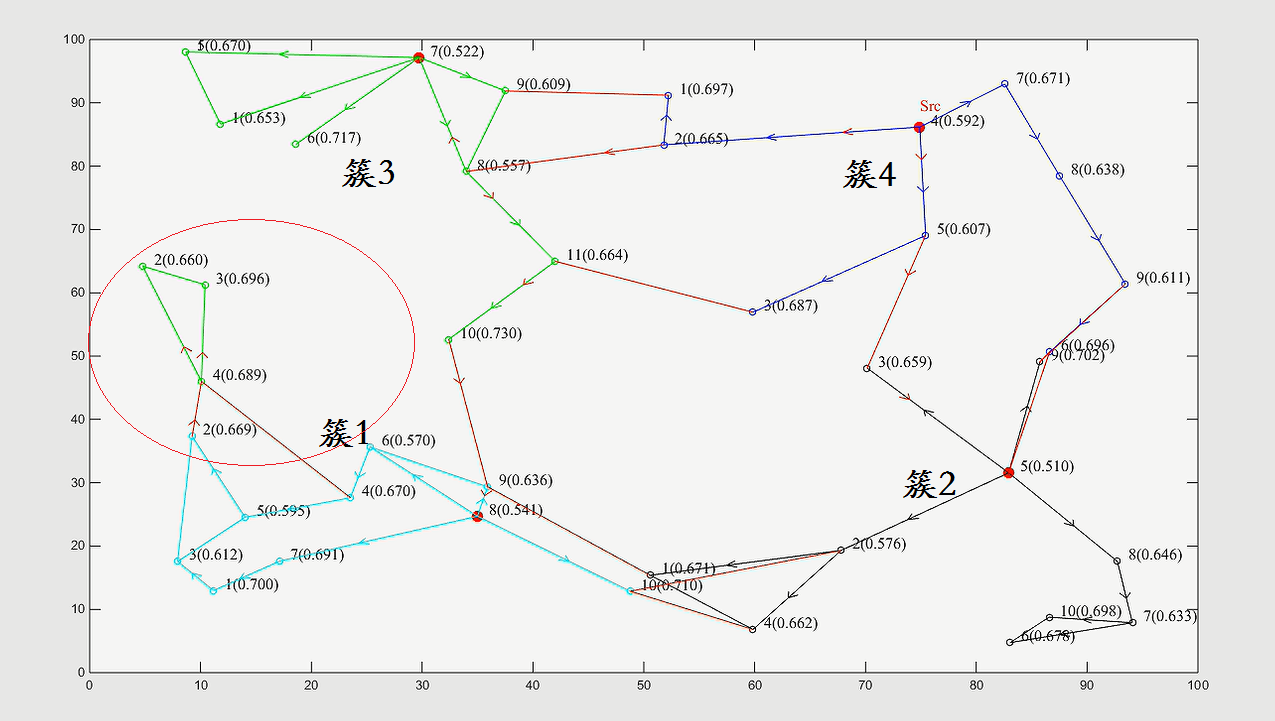


图2.1 与簇首脱离的节点

1. 组播树构建

所有推举出的簇首构建组播树，簇间成员的通信依靠簇首来完成。随机选择某一簇首为组播源，将信息组播至各个簇首。

src

图3.1 组播树—虚链路

如图3.1所示，簇首之间通过构建虚链路，形成组播树。所谓虚链路的含义是，假定簇首之间存在某一条路径，使得簇首可以互通。

考虑实际的情况，为减少组播树中的冗余路径，组播树中可能会存在一些分叉点。

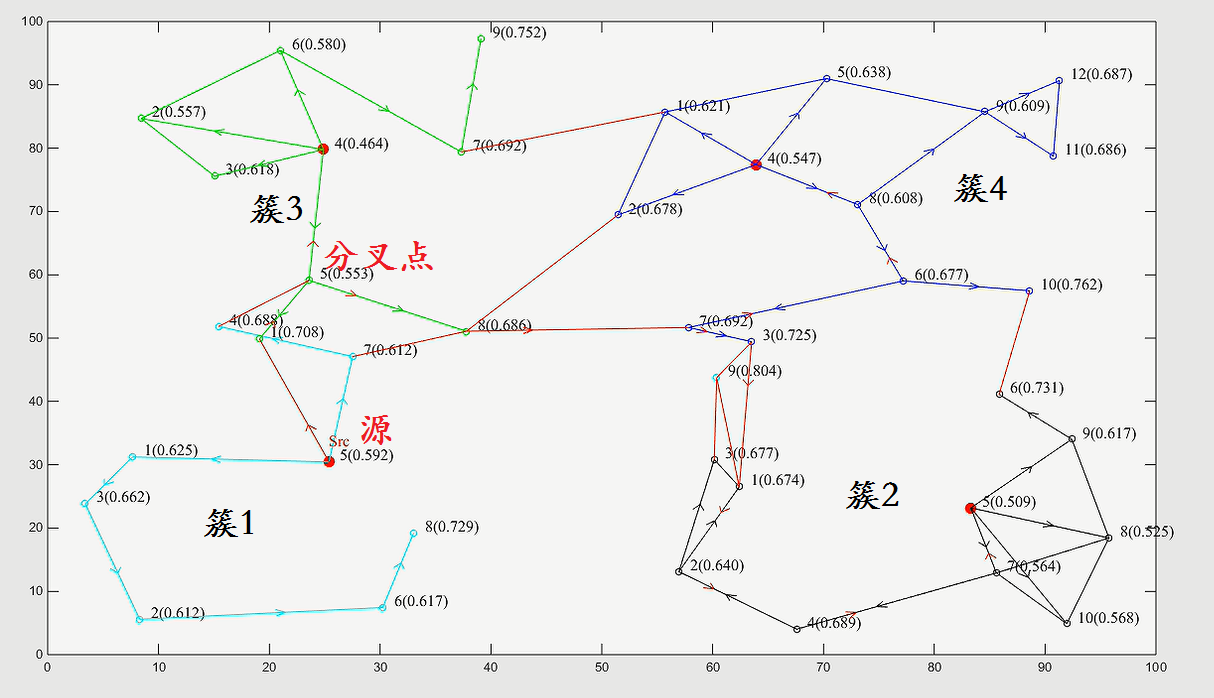


图 3.2 组播树—分叉点

如图3.2所示，若节点根据其所在的位置，分成图中所示的四个簇，并选定簇1的节点5作为簇首，则簇1至簇3的组播路径为5(簇1)-->1(簇3)-->5(簇3) -->4(簇3)，而簇3至簇4的组播路径，若还是以簇3的簇首（即节点4）为起点，则其组播路径为4(簇3)-->5(簇3)-->8(簇3)-->2(簇4)-->4(簇4)。可以看出，这条路径与簇1至簇3的组播路径有重合，即产生了冗余路径4(簇3)-->5（簇3）。为了减少组播树的冗余路径，簇1至簇4的合理组播路径应为图中所示的组播路径，即：

5(簇1)-->1(簇3)-->5(簇3) -->4(簇3)

8(簇3)-->2(簇4)-->4(簇4)

其中簇3的节点5就是一个分叉点。

* 1. 建立组播树过程

随机选择某个簇首作为组播源，记为src。则建立组播树的过程可以分解为（括号中s,p,q表示节点所属的簇）：

1. src(s)-->mediator(s)-->mediator(p)。标记簇s已被访问
2. mediator(p)-->head(p)。标记簇p已被访问。
3. head(p)-->mediator(p)-->mediator(q) or mediator(p)->mediator(p)-->mediator(q)。
4. 重复步骤2和步骤3，直至所有簇均被标记已访问。
   * 1. 组播源与中间节点

建立组播树的第一步是在组播源src和其所在簇的mediator之间建立路径。在前文中，已经建立了组播源至簇内其它节点的信息传输路径。如果将某个mediator选为组播树的一部分，则组播源至该mediator的信息传输路径也就作为组播树的一部分。若有多个mediator与不同的簇的相通，则组播源与所有的这些mediator之间的簇内传输路径都应属于组播树的一部分，也就是说组播源应尽可能地将信息扩散至所有与其可达的相邻簇。

一个簇中可能同时存在多个mediator，若组播源所在的簇中的多个mediator的链路的另一端的节点同属于一个簇（记为簇p），此时的准则为：若组播源与某个mediator之间的路径跳数最少，则选择该mediator。若跳数相同，则优先选择与簇p的簇首直接相连的mediator，若不存在这样的mediator，则选择传输时延最小的mediator。由于应用层组播节点的移动性，簇首可能不能总是与其它的节点都保持可连通，此时若所有的mediator都与组播源不可达，则组播树就无法成功地构建。

组播源src与其所属簇的mediator之间的组播树分支构建完成后，若mediator的中间链路仅有一条，则将这条中间链路添加进组播树中。若mediator的中间链路不仅一条，且所有中间链路的另一端节点均属于同一簇（记为簇p）时，此时，应优先选择与簇p簇首直接相连的mediator，若不存在这样的mediator，则选择链路传输时延（即节点间距离）更小的mediator。最后，标记组播源src所在的簇s已被访问。

* + 1. 中间节点与簇首

组播源src与中间节点mediator的组播树分支构建完成后，需要构建相邻簇（记为簇p）的mediator与其簇首的组播树分支。同样，这一组播树分支的构建也是基于簇内传输路径。若mediator与簇首之间存在一条簇内传输路径，则将该路径作为组播树的一部分。由于节点的移动特性，可能存在mediator与簇首不可通的情况，则此时通过该mediator构建簇p的组播树分支就会失败，需要从簇p的其它mediator寻找构建组播树分支的机会，若所有mediator均不能与簇首相通，则簇p的组播树分支就会构建失败，组播树的分支在簇p就停止向外延伸了。前文中已经描述过，为了使簇中与簇首脱离的部分尽可能地能够接收其它簇的信息，若脱离的部分中包含mediator，则其中间链路的另一端的节点在接收到簇收发送的信息后，会将信息也转发给mediator，mediator再将信息转发给与其相通的其它节点。

下一步需要构建的则是簇p与其相邻簇(记为簇q)之间的组播树分支。上文已经描述过，为减少组播树的冗余路径，组播树中可能存在分叉点。簇p与簇q之间的组播树分支有两种可能：（1）若簇p的簇首同时也是一个与簇q相连的mediator，则不存在分叉点，即此时的组播树分支为mediator(p) -->head(p)--> mediator(p)-->mediator(q)。（2）若簇p的簇首不是一个mediator，则此时会存在分叉点，即此时的组播树分支会分为两个部分: mediator(p)--> head(p)及mediator(p)->mediator(p)--> mediator(p)-->mediator(q)。(括号中p、q表示节点属于的簇)

当簇首同时也是一个mediator，即不存在分叉点时，类似的，若簇首的中间链路仅有一条，则将该链路加入组播树中；若簇首的中间链路不仅一条，则选取链路的准则为：优先选择与簇q的簇首直接相连的链路，若不存在这样的链路，则选择传输时延最小的中间链路。

当簇首不是一个mediator，即存在分叉点时，则需要构建mediator (p)至另一mediator(p)的组播树分支，其中第二个mediator(p)是与簇q相连接的。在本文中，基于Dijkstra算法，以传输时延为权重，构建了该组播树分支。中间链路mediator(p)-->mediator(q)的选取准则与上文描述的一致。最后需要标记簇p已被访问。

不断地重复以上的过程，直至所有的簇均被标记已访问，则组播树构建完成。

1. 组播树的动态维护

网络中的所有节点需要保存一份节点列表nodeList，nodeList包含两个信息：同一簇内其它节点的IP地址及与该节点是否可达（可达记为1，不可达记为2），nodeList用于记录与同一簇节点的通信状态。为了知悉节点是否处于工作状态以及节点间是否处于可通信状态，节点需要以时间T为周期，周期性地向其一跳邻居发送心跳包，若节点在收到某节点的心跳包，需向该节点发送响应报文，并将该心跳包转发至其一跳邻居。节点在重复收到同一节点的心跳包时，会丢弃多余的心跳包，并仅转发一次心跳包。若在3T时间内均没有收到簇内某个节点的心跳包，则可判定节点与该节点不可达。节点根据心跳包，周期性地更新nodeLsit。

由于网络时刻处于动态之中，簇内节点间的链路可能会因为节点间的距离过大，超出了节点的有效通信距离而断开，从而可能导致某些节点与簇内的其它节点无法通信。类似地，节点的移动也可能会导致某些中间链路的断开，影响簇与簇之间的正常通信。同时，考虑节点的带宽是有限的，因此需要为每个节点添加度约束，本文中实现度约束的方式是为每个节点设定一个最大的出度。节点的度约束对于维护节点间的正常通信也会有影响。

由于节点的移动具有一定的随机性，节点可能会移动至其它簇的区域内，此时节点可能因距离原因，无法与其所属簇内的节点正常通信。节点也有可能因为设备故障或者是功率管理等原因崩溃或者离开，节点的离开可能会导致某些正常工作的节点无法互相通信。在某个时间点，可能会有新节点加入某个簇，为了使新节点能与簇内的节点建立通信，需要采取措施。基于以上的描述，可见需要制定一些策略以增强应用层组播的稳定性和强壮性，保证簇内节点间以及簇与簇之间的正常通信。

* 1. 节点重连策略

正如上文中提及的，节点间的距离超出节点的通信范围以及节点的度约束，均会影响簇内节点和簇与簇之间的正常通信，本文提出节点重连策略，以加强应用层组播树的稳定性和强壮性。节点重连策略优先保证簇内节点间的正常通信，其次保证簇间的正常通信。

首先讨论以下两种可能出现的场景：

场景1：簇p的某节点n，依赖节点m来与簇内其它节点保持可通信状态，但节点m的度已经是最大值。

处理策略：

* + 1. 尝试断开节点m与簇内其它节点的某条链路，若该链路的断开，不会影响节点m及簇内其余节点的正常通信，则确认断开该链路，称这种链路为“无影响簇内链路”。若节点m不存在无影响簇内链路，则采用策略(2)。
    2. 尝试断开节点m的某条中间链路。若节点m存在多条中间链路，并且断开某条中间链路不会影响簇与簇之间的正常通信，则断开该链路，称这种链路为”无影响中间链路“。若节点m不存在无影响中间链路，或者节点m仅有一条中间链路，则无论是否会影响簇间的通信，都断开该链路，这种做法是为了优先保证簇内节点间的正常通信。

n

k

m

图 4.1 场景1

如图4.1所示，节点n因与其它节点的距离过大，而只能与节点m尝试建立链路，则节点n依赖与节点m的链路来保持其与簇内其余节点的通信，若节点m的最大度为3，则因为度约束，节点m无法与节点n建立链路，此时节点n应断开与节点k的链路，再与节点n建立链路。与节点k的链路断开，不会影响节点n及簇内其它节点的正常通信。

场景2： 节点n需要与其相邻簇p的节点m相联，才能保证簇间正常通信，而节点m的度已达最大值。

处理策略：

节点m应尝试断开其某一条链路（簇内或中间链路），若断开该链路后，不会影响节点m、簇内其它节点及簇间的正常通信，则确认断开该链路，并与节点n建立链路。

n

k

m

图 4.2 场景2

如图4.2所示，若节点n须与节点m建立链路，才能保证簇间的正常通信，而节点m已达最大度3，则节点应断开与节点k的链路，与节点n建立链路。与节点k断开链路，并不会影响节点m及其簇内的其余节点的正常通信。

节点根据其收到的心跳包，检测是否与同一簇内的某个节点或者某些节点不可达。若簇内存在节点不可达的某个节点，则节点应向簇内的所有非1跳邻居发送链路搭建请求报文。簇内的节点根据自身的最大节点度和出度判断，是否可以与该节点搭建链路，分为以下情况：

1. 若节点的出度小于最大节点度，则向该节点响应链路可搭建报文。
2. 若节点的出度等于最大节点度，基于以上讨论的两种场景，采取相应的策略，尝试断开节点的某条链路，若链路断开成功，则向该节点响应链路可搭建报文。

节点发送的链路可搭建报文中，应包含该节点的nodeLsit。当节点收到链路可搭建报文后，与相应的节点搭建链路，链路搭建成功后，根据链路可搭建报文中附带的nodeList以及自身的nodeList，判断是否能与同一簇内的所有其它节点可达，分为以下情况：

1. 节点能够与同一簇内的所有其它节点通信，节点重连成功，节点重连过程结束。
2. 节点仍不能够与同一簇内的所有其它节点通信，此时节点应向与其搭建新链路的节点发送重连报文，节点收到重连报文后，采取节点重连策略，重复以上过程。

由于节点可能向多个节点发送了链路搭建请求报文，因此，可能会有多个节点响应链路可搭建报文，则节点在接收链路可搭建报文时有多种状态：

1. 若该节点在接收到链路可搭建报文时，已经与簇内的某一节点成功搭建了新链路，则返回链路已搭建报文，节点接收到已搭建报文后，停止搭建新链路。
2. 若节点在接收到链路可搭建报文时，正在与簇内的某一节点搭建新链路，则返回链路正在搭建报文，节点在接收到该报文后，等待一段时间重新发送链路可搭建报文。
3. 若节点在接收链路可搭建报文时处于空闲状态，则向该节点发送响应报文，并搭建新链路。

节点向簇内节点发送链路搭建请求报文时，若节点与簇内的所有其它节点距离过远，超出节点的通信范围时，则节点无法采取重连策略，与其它节点重新建立连接。

本文的应用场景是战术通信网，战术通信网中的节点通常具备卫星通信的功能，若节点因移动导致与某个邻居节点的链路断开，导致该节点与簇内的大部分节点不可达，节点可以向卫星发送节点列表nodeList（包含同一簇节点的IP地址），卫星收到节点列表后，计算出该簇节点的“质心”坐标（即将簇中节点的坐标求均值所得的坐标值），卫星将该质心坐标发送给节点，节点收到后，往质心方向移动，以增大重新建立链路的机会。需要注意的是，节点往质心方向移动，可能会使某些中间链路的断开，影响簇间的正常通信，这种情况的出现是因为簇内节点间的距离过小，节点过于密集导致的，因此，在卫星检测到某个簇内的节点间距离方差（即节点的坐标与质心坐标的方差）小于某个门限值时，应通知该簇内的节点往质心的反方向移动，以恢复簇与簇之间的正常通信。

在战术通信网的场景下，认为节点一般不会频繁地离开簇的区域，若节点移动至区域外，应通过卫星使节点往所属簇的质心方向移动。但是，若节点移动至其它簇，且节点与所属簇内的大部分节点连接不上，应尝试与节点此时所在区域范围内的簇内节点相连，以尽可能地接收到其它簇节点的信息。

* 1. 节点的离开与加入

节点的离开事件与新节点的加入事件，应以什么样的频率发生？（赋予节点生存概率的概念）

* 1. 簇的分裂与合并

参考文献

[1] W. Su, S.-J. Lee, M. Gerla, “Mobility prediction in wireless networks,” MILCOM 2000, Volume: 1, Pages:491-495, 22-25 Oct. 2000

[2] 杨玲.基于优先度的层次化应用层组播算法研究[D].华中科技大学,2007.

[3] Yao Yu, Qi Zhang, and Sidan Du,“Mobility-aware Multicast over Mobile Ad Hoc Networks,” in IEEE International Colloquium on Computing, Communication, Control, and Management, CCCM 2009.