

基于主干子图的混合布局算法

张伟明, 张 凯, 王清贤

(信息工程大学 信息工程学院, 郑州 450002)

(amzw@tom.com)

摘 要:基于主干子图理论,提出了一种能够对幂率特征图进行布局的混合布局算法,其基本思想就是将待布局的原始图分解为主干子图和若干桩树,采用不同的布局算法对其进行分别布局。实验结果表明,当图的规模小于一定常数时,算法性能要优于传统的 K-K 算法,且在布局效果上,能使用户较容易地区分出图中的主干子图和各桩树。

关键词:绘图;主干子图;幂率;布局

中图分类号: TP393.08 **文献标志码:** A

Hybrid layout algorithm based on skeleton subgraph

ZHANG Wei-ming, ZHANG Kai, WANG Qing-xian

(College of Information Engineering, Information Engineering University, Zhengzhou Henan 450002, China)

Abstract: We presented a novel hybrid layout algorithm based on skeleton subgraph, which could handle the power-law graph. The key idea was to decompose the original graph into a skeleton subgraph and several stub trees, and to layout them with different graph drawing algorithms. The experiments and analysis indicate that our algorithm outperforms the traditional K-K algorithm when the size of the graph is smaller than a certain constant, and the result seems to be easier to lead the user to identify the skeleton subgraph and the stub trees, and to understand the original graph.

Key words: graph drawing; skeleton subgraph; power law; layout

0 引言

基于主干子图的混合布局算法(Skeleton-based Hybrid Layout Algorithm, SHLA)有效利用了多数复杂网络所具备的幂率特征,对主干子图采用改进后的三维 K-K 算法,而对桩树节点采用插值算法,将其呈扇形分布在桩树根节点的周围,从而在保证较好布局效果的前提下,提高了图的整体布局效率。

1 相关工作

1.1 力导向图布局算法

力导向图布局算法由文献[1]提出,其基本思想是模拟力学平衡原理,将图中的节点模拟为钢环,连接则模拟为弹簧,绘图的过程则是不断调整钢环的位置,最终达到力学平衡状态。Kamada 和 Kawai^[2](简称 K-K 算法)在以上基础上,引入了胡克定律作为系统能量计算的基本依据。由于 K-K 算法易于理解,且便于实现,这使得 K-K 算法成为力导向图布局算法中最重要的算法之一。本文的主干子图布局算法就是在 K-K 算法的基础上加以改进得到的。

1.2 混合布局算法

混合布局算法的基本思想是将布局过程拆解为不同的步骤或将布局区域分块,然后在不同步骤、不同布局区域采用不同的布局算法,以求在整体上和局部上都尽可能地满足约束条件。由于采用了分治策略,混合布局算法总体上降低了问题规模,其布局效率也往往优于单一的布局算法,如 VLNT^[3]和 RDC^[4]。文献[5]将幂率特征图称之为全局图,该图是通过图中潜在的局部图不断相加而得到的,而局部图则是指局

部高度互联的图分量。它们以子图的大小和边的长度为依据将图的边分成局部边和全局边,并在此基础上给出了计算局部/全局划分的近似算法,得到图的划分后再用力导向布局算法对其进行布局^[6]。

1.3 主干子图及其相关性质

主干子图是定义在幂率特征图上的一个特殊子图,可以通过对原始图进行反复的悬挂点过滤得到,同时也可识别出附着在主干子图上的多个桩树,如图 1 所示。主干子图是一些度相对较高、数量相对较少节点的集合,而桩树则正好相反,幂率特征有效地保证了节点度分布的这一非均一特性。

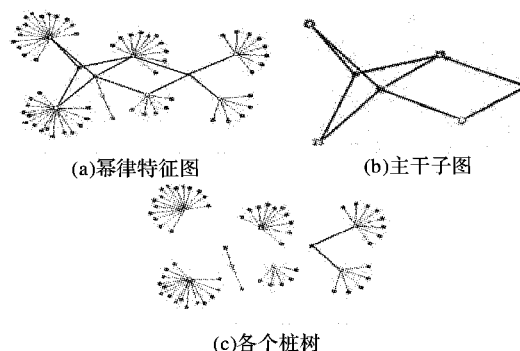


图 1 幂率特征图的划分

由于主干子图理论不是本文讨论的重点,在此不再赘述。本文不加说明地使用了主干子图生成算法 `sketelon_identify` 和桩树生成算法 `grow_stub_tree`。其中 `sketelon_identify` 的输入为原始图,输出为主干子图;`grow_stub_tree` 的输入为主干子图上的桩树根节点,输出为对应的桩树。

收稿日期:2007-09-04;修回日期:2007-11-10。 基金项目:国家 863 计划项目(20031AA146010)。

作者简介:张伟明(1978-),男,江西余干人,博士研究生,主要研究方向:网络安全; 张凯(1979-),男,江苏南京人,硕士研究生,主要研究方向:网络安全; 王清贤(1960),男,河南郑州人,教授,博士生导师,主要研究方向:算法设计与分析、网络安全。

2 基于主干子图的图布局算法

2.1 算法描述

美学标准^[7]是衡量布局算法优劣的重要标准之一,我们在设计 SHLA 算法时主要考虑了以下美学因素:1)区分主干子图和桩树;2)节点分布尽量均匀;3)尽量避免连接交叉。

对于美学因素1),由于对主干子图和桩树布局时采用了不同的布局算法,其布局风格和布局区域有显著区别,这将有助于用户直接识别出图中的主干子图和桩树;对于美学因素2),可以通过布局前计算桩树所需布局区域的方法合理利用布局空间,从而达到总体均匀的效果;对于美学因素3),我们将连线交叉检测的结果作为参数之一,考虑在节点能量的计算公式中,从而达到修正节点坐标的目的。

综合以上因素,基于主干子图的图布局算法可描述如下:

算法1 SHLA (G)

- 1) initialize (G)
- 2) $G_s(V_s, E_s) \leftarrow \text{skeleton_identify}(G)$
- 3) $\text{KK_EX_Layout}(G_s)$
- 4) FOR ALL $v \in V_s$
- 5) $\text{Tree}_C(v) \leftarrow \text{grow_stub_tree}(v)$
- 6) $\text{Stub_Tree_Layout}(\text{Tree}_C(v))$
- 7) END FOR

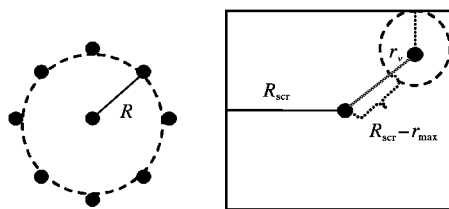
算法首先利用 skeleton_identify 识别出图的主干子图,并采用 KK_EX_Layout 算法布局;然后遍历主干子图上的所有节点,利用 grow_stub_tree 识别出所有的桩树,并采用插值算法 Stub_Tree_Layout 逐一为之布局。该算法主要包含两个子算法:主干子图布局算法 KK_EX_Layout 和桩树布局算法 Stub_Tree_Layout ,下面分别予以描述。

2.2 主干子图布局算法

主干子图布局算法主要是在 K-K 布局算法的基础上,综合利用前期改进的成果而得到,主要考虑的改进方法有:1)改善初始布局;2)改进单个节点坐标调整过程以加快算法收敛速度;3)改进节点能量值计算方法,从而减少连线交叉;4)改变算法的终止条件以加快算法收敛速度。

2.2.1 加权环形预布局算法

K-K 算法常用的初始布局算法是环形布局算法。该算法在一个半径为 R 的圆环上对 N 个节点进行布局,每个节点占据 $(2\pi/N)$ 度。这样虽然简单,但是并没有考虑任何输入的图的结构特征。而在 SHLA 算法中,由于第一阶段的 K-K 布局是针对主干子图进行的,而且主干子图中的每个节点都有不同的半径(桩树中包含的节点数不同,故占用的布局面积相应不同,由于桩树布局采用的是圆形布局风格,我们对主干子图中的节点布局区域统一用“半径”描述),所以不能等同对待,加权环形预布局算法就是针对这种情况设计的。



(a) 环形布局示意图 (b) 加权环形布局示意图
图2 环形布局和加权环形布局示意图

设 R_{scr} 是可布局面积允许的最大坐标值, r_v 是节点 v 的半径, r_{max} 是主干子图的所有节点中最大的半径,则有 $R = (R_{scr} - R_{max}) + r_v$ 。如图2(图(a)是环形布局示意图, R 取固定

值;图(b)是加权环形布局示意图, R 由计算得出)。

2.2.2 单个节点坐标调整算法的改进

文献[8]的 EL(Edge Length)算法在每次迭代时,都以 P 概率把节点 n 的坐标调整到其理想边长位置的平均点上。实验证明,这样可以加速 K-K 算法的收敛速度,并减少扭曲。我们在主干子图布局算法中直接使用了该方法。

2.2.3 单个节点能量值计算方法的改进

K-K 算法用能量值 E 来度量布局的平衡状态,所以可以通过修改能量计算公式来加入新的考量因素。为尽可能地避免边的交叉,我们对单个节点 m 的能量值计算公式调整如下:

$$\Delta_m = (1 + W_c C_c) \sqrt{\left(\frac{\partial E}{\partial x_m}\right)^2 + \left(\frac{\partial E}{\partial y_m}\right)^2 + \left(\frac{\partial E}{\partial z_m}\right)^2}$$

其中 W_c 是边交叉因素权重, C_c 是交叉边计数。在选择最大能量点时,如果节点 m 所关联的边与其他边交叉较多,则节点 m 的能量值较大,进而节点 m 更有可能被调整;在调整 m 坐标时,如果新坐标导致与节点 m 的交叉边增加,那么新的能量值会升高,新的坐标有可能不被接受。通过这种手段使 K-K 算法在布局过程中可以尽量减少交叉边。

显然,要计算节点 m 所关联的边与其他所有边的交叉数,时间复杂度至少为 $O(|E| \times |\text{neighbor}(m)|)$ 。我们把交叉边的检测设为可选,通过 W_c 来控制,如果 W_c 为0,则跳过交叉边检测。

单个节点能量值计算算法描述如下:

算法2 calculate_single_vertice_energy (m)

- 1) $E_m \leftarrow \sqrt{\left(\frac{\partial E}{\partial x_m}\right)^2 + \left(\frac{\partial E}{\partial y_m}\right)^2 + \left(\frac{\partial E}{\partial z_m}\right)^2}, C_c \leftarrow 0$
- 2) IF ($W_c > 0$) THEN
- 3) FOR ALL $e \in E_s$
- 4) FOR ALL $v \in \text{neighbor}(m)$
- 5) IF (边(m, v)与 e 交叉) THEN
- 6) $C_c \leftarrow C_c + 1$
- 7) END IF
- 8) END FOR
- 9) END FOR
- 10) $E_m \leftarrow E_m(1 + W_c)$
- 11) END IF

2.2.4 终止条件的改进

在兼顾布局效果的同时,为了提高主干子图布局算法的布局效率,防止振荡,我们设置了两个算法终止条件:1)总能量值的变化幅度小于1%;2)迭代次数达到节点数量的2倍。终止条件1表明各个节点基本达到了平衡位置,优化幅度已经很小,故可以结束算法;终止条件2则表明平均每个节点都有机会被优化两次,同时,由于节点数量是常数,故算法一定在若干次迭代以后结束,这就防止了由节点位置振荡导致算法不收敛的情形。

综上所述,主干子图布局算法 KK_EX_Layout 的基本流程可描述如下:

- 1) 加权环形预布局;
- 2) 计算图的全局能量值 E , 并选择能量最大的节点 P ;
- 3) 计算节点 P 的新坐标;
- 4) 采用 EL 算法调整节点 P 的坐标;
- 5) 计算 P 节点坐标调整后的图能量值 E_n ;
- 6) 判断是否符合算法终止条件,若符合,则成功退出;否则转入第2步。

2.3 桩树布局算法

当完成主干子图布局后,就需为桩树节点分配坐标。我们考虑的桩树节点坐标分配的基本原则是:1) 桩树节点和桩树根节点的连线应尽量与主干子图布局结果中的连线不交叉;2) 由于桩树节点为数众多,因此布局性能是我们考虑的重要因素。

针对原则 1,我们根据主干子图布局算法的结果,为每个桩树计算可布局区域,从而使这些桩树节点与其根节点的连线不可能与主干子图交叉。针对原则 2,我们采用插值算法将桩树中的其他节点插入到各个桩树根节点附近。

由此可见,桩树布局主要包含两个重要步骤:首先是确定桩树可用的布局区域;其次是找到合适的插值算法。

2.3.1 可布局区域的确定算法

按照定义,桩树的根节点一定是主干子图上的节点,所以各个桩树的根一定已经在前一阶段的主干子图布局算法中分配了坐标。各个桩树可以简单地环绕其树根而插入布局,但效果并不好,因为无法避免各桩树与主干子图交叠的现象,如图 3(a) 所示(实线为主干子图中的节点和连线,虚线为桩树节点和连线, A、B、C 是被主干子图中的连线所分割开的三个区域)。

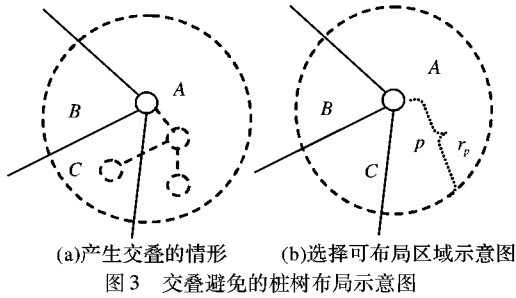


图 3 交叠避免的桩树布局示意图

显然,如果能够将桩树的布局区域控制在区域 A,则能有效避免图 3(a) 中的情形。如图 3(b) 所示, p 是一个桩树的根节点,由于在先前的初始化阶段计算了 p 的半径 r_p ,又在主干子图布局阶段考虑了 r_p ,所以一般来说,可以认为以 p 为圆心、以 r_p 为半径的圆内都是为 p 预留的可布局区域(图中用虚线的圆表示)。进一步观察可以发现,这个可布局的圆形区域又被 p 与其在主干子图上的邻居节点之间的边划分成 A、B 和 C 三个扇形区域,其中 A 区域最大,围成 A 区域的与 p 关联的两条边的夹角也最大。所以我们的算法目的就在于找到与 p 关联的、夹角最大的两条边,进而得到其起始角度和夹角角度。由此可见,可布局区域确定算法的基本可概括为,遍历与 p 关联的每一条边,求出其角度,并按从小到大排序,相邻的角度两两相减以求得其夹角,夹角最大的区域即为所求区域。算法描述如下:

算法 3 find_area_for_tree(r)

```

1)  $A \leftarrow \emptyset, angle_{\max} \leftarrow 0, angle_{\begin{smallmatrix} begin \\ end \end{smallmatrix}} \leftarrow 0$ 
2)  $N \leftarrow neighbor(r, G_s)$ 
3) FOR ALL  $v \in N$ 
4)  $A \leftarrow A \cup \{ \arctan(\frac{v_y - r_y}{v_x - r_x}) \}$ 
5) END FOR
6) sort(  $A$  )
7) FOR ALL  $a_i \in A, i = 0, 1, \dots, |A|$ 
8) IF (  $i = |A|$  ) THEN
9)  $angle \leftarrow a_i - a_0$ 
10) ELSE  $angle \leftarrow a_{i+1} - a_i$ 
11) END ELSE

```

```

12) END IF
13) IF (  $angle > angle_{\max}$  ) THEN
14)  $angle_{\max} \leftarrow angle, angle_{\begin{smallmatrix} begin \\ end \end{smallmatrix}} \leftarrow a_i$ 
15) END IF
16) END FOR

```

该算法也有一定的不足,就是当某一桩树的根节点 p 的可布局区域被划分成很多个扇形区域,而 p 所在的桩树节点被分布在多个子树上时,只选择最大的那个扇区并不能合理利用空间。解决的方法是按照 p 的子树的多少和大小来选择多个扇区,最大的子树占用最大的扇区,其他类推。

2.3.2 桩树节点坐标分配算法

接下来的工作就是在这个扇形区域内插入树上节点。对 n 层树,我们可以把布局扇区想像成被 $n-1$ 个同心圆穿过,最大的同心圆半径等于处于圆心的树根节点 p 的半径,其他圆的半径则依次递减($r_p/n-1$)。把每一层节点布在对应的同心圆上,其角度是其子孙中的叶子节点数量的函数,这样就能合理地分配节点位置。

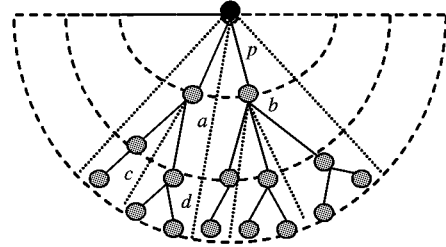


图 4 扇形区域内的树的布局算法示意图

以图 4 为例,对桩树根节点 p 来说,若可用的布局扇区为 $angle_p$,由于 p 的孩子节点 a 的子孙中有 3 个叶子节点, b 的子孙中有 5 个叶子节点,故 a 占用 $angle_a = \frac{3}{8}angle_p$, b 占用 $angle_b = \frac{5}{8}angle_p$ 。同理, a 的孩子 c 占用 $angle_c = \frac{1}{3}angle_a$, d 则占用 $angle_d = \frac{2}{3}angle_a$,其他节点皆可依此类推。

通过根节点的坐标、各个节点占用的角度和对应的同心圆半径,就可以计算出每一个桩树节点的 X 轴坐标和 Y 轴坐标。显然这是一个递归的过程,递归的起始条件为桩树根节点坐标和待布局区域,递归终止于桩树的最低层节点。

2.4 布局效果与时间复杂度分析

2.4.1 布局效果

图 5 是 SHLA 算法在实际应用中的效果。

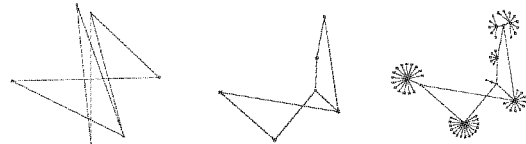


图 5 SHLA 算法布局效果

可以看出,最终的布局结果基本符合预期所考虑的美学因素。

2.4.2 时间复杂度分析

分析 SHLA 算法的最坏时间复杂度,若输入的图 G 具有 n 个节点,其主干子图 G_s 上有 s 个节点,则各步骤的时间复杂度分析如下:

- 1) 主干子图的生成。计算复杂度为 $O(n^2)$ 。
- 2) 所有桩树生长的过程。计算复杂度为 $O(n-s)$ 。
- 3) 主干子图上所有节点间最短路径的计算。采用

Johnson 所有节点间最短路径算法,其最坏时间复杂度为 $O(s^2)$ 。

4) 主干子图布局算法中的节点坐标调整。根据算法终止条件,对各个节点坐标调整的迭代次数最多不超过 $2s$;而每个节点新坐标的生成则需遍历整个节点集,故其执行次数最多为 $2s^2$,即最坏时间复杂度为 $O(s^2)$ 。

5) 主干子图布局算法中的全局能量值计算。根据算法2,由于考虑了边的交叉,主干子图上每个节点的能量值计算需要遍历所有的与之关联的边,每个节点所关联的边最多为 $(s-1)$ 条,对所有节点计算则需要执行 $s(s-1)$ 次;根据终止条件,其迭代次数最多为 $2s$,故其执行次数最多为 $2s^2(s-1)$,即最坏时间复杂度为 $O(s^3)$ 。

6) 所有可布局区域的识别。对于任意桩树的根节点,其周围可布局区域由所有与之相连的边所划分,所以其可布局区域的数量等于其度数;主干子图上每个节点最大度数为 $(s-1)$,所以执行次数最多为 $s(s-1)$,即最坏时间复杂度为 $O(s^2)$ 。

7) 所有桩树的布局。算法为递归算法,所以桩树的布局最坏时间复杂度与桩树生长过程相同,为 $O(n-s)$ 。

综上所述,整个 SHLA 算法的最坏时间复杂度为: $\max(O(n^2), O(n-s), O(s^2), O(s^3))$ 。由于 $s \leq n$,故可简化为: $\max(O(n^2), O(s^3))$ 。

若 $s = n/10$,则当 $n \leq 1000$ 时,SHLA 算法的最坏时间复杂度为 $O(n^2)$,而标准 K-K 算法的平均时间复杂度为 $O(n^3)$ 。

3 结语

本文基于主干子图理论,提出了一种能够处理幂率特征

图布局的混合布局算法 SHLA,其基本思想就是将待布局的原始图分解为主干子图和若干桩树,然后采用不同的布局算法对其分别进行布局。由于主干子图是少量度较高节点的集合,此时对主干子图采用改进后的 K-K 算法,在性能和布局效果上能取得较好的平衡。相比而言,为达到较好的布局效果,桩树布局算法的主要时间开销主要花在了选择可布局区域的过程中。

参考文献:

- [1] EADES P. A heuristic for graph drawing[J]. Congressus Numerantium, 1984, 42: 149 - 160.
- [2] KAMADA T, KAWAI S. An algorithm for drawing general undirected graphs[J]. Information Processing Letters, 1989, 31(1): 7 - 15.
- [3] AU S C, LECKIE C, PARHAR A, et al. Efficient visualization of large routing topologies[J]. International Journal of Network Management, 2004, 14(2): 105 - 118.
- [4] SAGIE G, WOOL A. EES2003-7, A clustering approach for exploring the Internet structure[R]. Electrical Engineering Systems, 2004.
- [5] ANDERSEN R, CHUNG F, LU L. Analyzing the small world phenomenon using a hybrid model with local network flow[C]// Proceedings of the 3rd Workshop on Algorithms and Models for the Web-Graph. Berlin: Springer, 2004: 19 - 30.
- [6] ANDERSEN R, CHUNG F, LU L. Drawing power law graphs using a local/global decomposition[J]. Algorithmica, 2007, 47(4): 379 - 397.
- [7] BATTISTA G D, TAMASSIA R, TOLLIS I G, et al. Algorithms for the visualization of graphs[M]. [S. l.]: Prentice-Hall, 1999.
- [8] RODGERS P, MUTTON P. Visualizing weighted edges in graphs[C]// Proceedings of the 7th International Conference on Information Visualization. [S. l.]: IEEE, 2003: 258 - 263.

(上接第 377 页)

从图 8 和图 9 信令开销对比图中,我们可以看出采用平台 IMS 服务器共享用户状态信息的优化方案时,因为新旧接入平台 IMS 服务器可以共享用户状态信息,所以,当进行重注册和会话重新建立时可以少发送很多个信令包,从而使用户终端与平台 IMS 服务器之间的 SIP 信令开销大大减少,尤其是在会话重新建立的过程中,大约减少了 66%。对于平台通信系统,用户较多的情况下,减少信令开销可以极大地提高系统的性能。

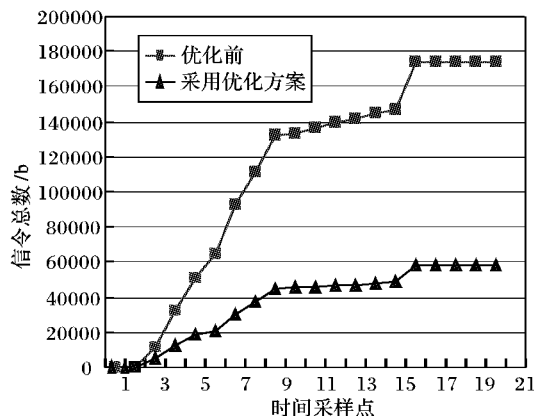


图9 平台 → UE 的信令总数量

4 结语

升空平台通信系统是现有通信网络向空间的延伸,是构

筑国家信息基础结构的又一重要组成部分。研究在升空平台通信系统中,用户终端发生跨平台移动时,怎样进行快速切换,减少切换时延,提高系统性能,具有很重要的意义。我们知道升空平台通信系统覆盖面比较大,传输距离比较远,用户数目也比较多。在这种情况下,发生切换时,过多的信令包的传输,会带来较大的切换时延和信令开销。本文采用通过新旧接入平台间共享存储在旧的接入平台 IMS 服务器中的用户状态信息来减少切换时的信令包的数目,从而达到减少切换时延的效果。

参考文献:

- [1] 樊昌信. 一种发展中的新移动通信方式: 平流层通信研发概况[J]. 现代电子技术, 2005, 28(19): 1 - 3.
- [2] 3GPP TS23. 228 v6. 9, IP Multimedia Subsystem (IMS) Stage 2 (Release6)[S], 2005.
- [3] BANERJEE N, ACHARYA A, DAS S K. Seamless SIP-based mobility for multimedia applications[J]. Network, IEEE, 2006, 20(2): 6 - 13.
- [4] SOLOMON J D. 移动 IP[M]. 裘晓峰译. 北京: 机械工业出版社, 2001.
- [5] RFC3261, SIP: Session initiation protocol[S], 2002.
- [6] LARSEN L, MATTHIESEN K V, SCHWEFEL E, et al. Optimized macro mobility within the 3GPP IP multimedia subsystem[C]// International Conference on Wireless and Mobile Communications. Bucharest: IEEE, 2006: 82.