基于禁位排列原理的路由决策算法*

史 岚，吕建辉
（东北大学 信息科学与工程学院，沈阳 110819）

摘 要：针对已有的 OSPF 路由选择协议作进一步的研究，分析并运用了数学原理中的禁位排列方法，设计出一种新的路由决策算法。研究表明，路由单链排列状态下，该算法与 Dijkstra 算法具有相同的时间复杂度；在实际网络结构下，存在算法的时间复杂度处于 $O(n^2)$ 到 $O(n^3)$ 之间或者该算法在时间复杂度是 $O(n^3)$ 时具有良好的优越性。网络拓扑结构改变，该算法具备较好的收敛性，说明该算法拓展了 OSPF 并能够应用到广泛的领域。

关键词：OSPF；禁位排列；路由决策算法；时间复杂度；收敛性
中图分类号：TP391 **文献标志码**：A **文章编号**：1001-3695(2014)01-0257-04
doi:10.3969/j.issn.1001-3695.2014.01.060

Routing decision algorithm based on limit arrangement

SHI Lan, LV Jian-hui
(*College of Information Science & Engineering*, *Northeastern University*, *Shenyang* 110819, *China*)

Abstract: In order to do further research on routing protocol of OSPF, this paper analyzed and introduced the method of limit arrangement in mathematical principle, and designed a new routing decision algorithm. It showed that this algorithm and Dijkstra algorithm were the same in time complexity when the arrangement of routing was under the condition of single strand. Meanwhile it existed that the time complexity was between $O(n^2)$ and $O(n^3)$, and it could possibly be a good superiority when the time complexity was $O(n^3)$ when the arrangement of routing was in the actual network structure. When the network topology changes, the algorithm always has a good convergence, which show the algorithm expands OSPF and has widely application.

Key words: OSPF; limit arrangement; routing decision algorithm; time complexity; convergence

因特网的内部网关协议主要包括 RIP 和 OSPF 两种不同的协议[1]。RIP 协议是一种分布式的基于距离向量的路由选择协议，而 OSPF 协议是为了克服 RIP 协议在 1989 年开发出来的，它是基于最短路径算法的一种协议。然而 RIP 协议有个最大的缺点就是遇到坏消息的时候，传播得很慢。为了避免或者弱化这一弊端，不少科研人员采用了诸多的方式进行改进，比如让路由器记录收到某特定路由信息的接口[2]，将其他优化算法结合到该路由协议之上。尽管如此，这一弊端依然存在。OSPF 协议虽然克服了 RIP 协议的“坏消息传播慢”这一特点，但由于它要求每个路由器都要知道网络的拓扑结构，因此在计算各个路径的代价时，必然开销大量的时间。一旦网络拓扑发生改变，每个路由就必须重新参与算法，这一环节的收敛快慢直接影响着网络的性能，因此关键的问题就放在了各个路由表的更新快慢上。而路由表的更新快慢一是取决于网络的拓扑复杂程度，二是取决于各个路由表信息的交互量多少。越是复杂的拓扑结构，在信息的交互过程中相应地就越复杂，时而会出现信息重复发送情况，造成时间浪费；路由携带的信息越多，在交互的时候网络的承载压力就会越大。

为了在不减少信息交互量的同时去缩减网络压力，人们对路由信息传播策略开始进行深入化、层次化研究，越来越多的学者倾向于容迟网络算法、蚁群算法、神经网络算法[3~5]等小型应用型算法。尽管这些算法在原理上有所区别，但其中的基本模型都是基于数学原理展开的。网络拓扑结构本身就是一个复杂的组合数学问题，同样可以在这一层面上入手，继续探讨相关的路由决策算法。

1 禁位排列

1.1 路由决策思想

组合论中提到基于错位排列[6]的其他禁位问题[7]。错位排列简单描述为：集合 $S=\{1,2,\cdots,n\}$ 的一个排列 $i_1,i_2,\cdots,i_n$ 称为它的一个错位排列。然而基于错位排列的其他禁位问题却有所不同，它要求任何一次排列 $i_1,i_2,\cdots,i_n$ 中不能出现 12，23，…，$(n-1)n$ 这样的形式。在网络拓扑结构中，其中的每个路由相当于集合 S 中的一个数字，而各个路由（每个路由有自己的编号）之间的排列相当于初始化的集合 S。在初始化阶段，相邻路由之间直接交互信息（不经过任何跳数），并把得到的信息保存在自己的路由表中，当经过一次禁位排列之后，路由将重新认知另外的路由，最大化地避免了重复路由之间的信息重复交换。这样经过多次禁位排列、路由交换之后，生存在网络拓扑结构中的 n 个路由就互相认知了消息，从而形成了信息的一致性。这种原理的提出与下面的理论密不可分：

a）网络拓扑结构[8]就是一个图，要想使各个路由消息互换，从任一路由节点出发都能找到一个连接各个路由的通路，这样的通路可看做一个带有编号路由的一次排列。

收稿日期：2013-04-07；**修回日期**：2013-05-18 **基金项目**：国家自然科学基金资助项目（61100182）

作者简介：史岚（1964-），女，辽宁沈阳人，副教授，硕士，主要研究方向为计算机网络与信息安全（shilan@mail.neu.edu.cn）；吕建辉（1988-），男，硕士研究生，主要研究方向为计算机网络与信息安全.

b)相邻路由直接交互,消息传递得较快,因此要最大化构造这样的相邻路由。

c)实际的网络拓扑结构图中,或多或少地存在环路,每增加一个环路,将会减少禁位排列的次数,收敛的速度更快。

1.2 最优禁位排列选择

路由初始化的编号均是顺序排列的正整数。经过一次禁位排列之后,得到的排序不能出现 12,23,…,$(n-1)n$ 和 21,32,…,$n(n-1)$ 两种模式,一旦出现,即说明路由器要重复传递信息。当模式 1 和 2 相互独立时,根据容斥原理[9]进行推导,其排列种类为 N。

$$N = n! - \binom{n-1}{1}(n-1)! + \cdots + (-1)^{n-1}\binom{n-1}{n-1}1! \quad (1)$$

然而,模式 1 与 2 混合排列,如 12、43 形式的路由编号出现时,也不符合禁位排列的要求。因此在计算总的禁位排列种类时,必须要兼顾模式 1 与 2 的情况。基于这种思想,先选择模式 1 进行禁位排列,得到数目 N;然后剔除其中包含的模式 2 中的形式;最后得到禁位排列的总数。例如路由初始编号为 $\{1,2,3,4,5\}$,其禁位排列如下:

$\{1,3,5,2,4\}$ $\{1,4,2,5,3\}$ $\{2,4,1,3,5\}$ $\{2,4,1,5,3\}$ $\{2,5,3,1,4\}$
$\{3,1,4,2,5\}$ $\{3,1,5,2,4\}$ $\{4,2,5,3,1\}$ $\{3,5,2,4,1\}$ $\{5,3,1,4,2\}$
$\{3,5,1,4,2\}$ $\{4,1,3,5,2\}$ $\{5,2,4,1,3\}$ $\{5,3,1,4,2\}$

对每一初始化连续编号的路由,第一次禁位排列选择的种类为 Q_n。然而,所得到的 Q_n 个排列并非都是最优的,因此选择一个最优的禁位排列对以后的高效路由决策算法的执行具有巨大的支持作用,最优排列事件为 P。选择 Q_n 中的任意两个排列 $\{i_1,i_2,\cdots,i_n\}$ 和 $\{j_1,j_2,\cdots,j_n\}$,分别称为事件 X、事件 Y,其数学期望为

$$E(X) = \sum_{k=1}^{n} |i_k - k| \quad (2)$$

$$E(Y) = \sum_{k=1}^{n} |j_k - k| \quad (3)$$

若 $E(X) \neq E(Y)$,则事件期望值较小的排列为最优的;若 $E(X) = E(Y)$,则判断 $D(X)$ 与 $D(Y)$,事件方差较小的排列为最优的。因为消息传递过程中,所经过的路由跳数相对平稳,不会影响整体的速度。如初始路由编号为 $\{1,2,3,4,5,6\}$,$Q_n = 14$,任意选择两个禁位排列,$X = \{1,3,5,2,4,6\}$,$Y = \{1,4,6,2,5,3\}$。$E(X) = 6$,$E(Y) = 10$。显然选择事件 X 型禁位排列较优。通过上述说明,选择最优禁位排列的过程,用伪代码描述如下:

```
Initially: V0 = {1,2,…,n}
//V0 is the arrangement of initial routing number
1 To calculate the number of limit arrangement without mode 1
  and then
  N: = the number of limit arrangement without mode 1.
  V: = V1 ∪ V2 ∪ … ∪ VN
2 from k = 1 to N
  if Vk contains mode 2
  V: = V/Vk//excluding Vk from V
  and then
  Qn: = the number of limit arrangement without mode 1 and mode 2
3 from i,j = 1 to Qn
  select Vi,Vj from V
  X: = Vi, Y: = Vj
  if E(X) < E(Y)        P: = X
  else if E(X) > E(Y)   P: = Y
  else if D(X) < D(Y)   P: = X
  else  D(X) > D(Y)     P: = Y
  P is the optimal limit arrangement
```

2 基于禁位排列原理的路由决策算法

2.1 算法分析

在网络拓扑中,一定能够找到一条连接各个路由的通路,并且这一通路中有多个分支,每个分支等同于一个顺序编号的路由排列。初始状态,同一分支上的路由两两相邻交互信息,而不相邻的路由则不能直接交互。那么,可以使初始化后的路由编号进行多次禁位排列,以达到所有的路由都进行两两直接交互。这样,整个分支上的路由信息便达到一致。值得指出的是,这种变换路由位置并不是真的将网络拓扑中的所有路由进行重置,而是虚拟的。

从最优禁位排列开始,构造一个类似对称布尔矩阵的 0-1 表,该表是关于主对角线对称的表格,表中元素记为 $a_{i,j}$,其中 $a_{i,j} = a_{j,i}$。路由编号对应相等,即路由知道本身的存储消息,记为 1;路由初始化,相邻路由直接进行交互消息,记为 1。初始化后,各个路由消息互传状态如表 1 所示。

表 1 初始化下的路由消息互传状态

ID	1	2	3	…	$n-1$	n
1	1	1	0	…	0	0
2	1	1	1	…	0	0
3	0	1	1	1	0	0
⋮	⋮	⋮	1	⋮	⋮	⋮
$n-1$	0	0	0	⋱	1	1
n	0	0	0	…	1	1

经过最优禁位排列,路由位置变化,新的相邻路由编号进行消息互传,记为 1;其他路由之间不曾交互消息的记为 0。在以后的每次排列中,采取深度优先遍历[10,11],每遍历一次排列,本轮相邻编号的路由能够进行消息互传,将 0 置为 1。当深度遍历的次数达到一定的阈值时,有些路由编号开始脱落,即与其他任何路由已经互通过消息(路由编号整行或整列上元素全为 1);这时,对那些脱落的路由编号不能再进行深度优先遍历,而是将其随机成链,连接到其他还未脱落的路由编号的深度优先遍历序列之后。这样做避免对脱落的路由再次遍历。当表中不再出现 0 时,说明所有的路由消息一致。为了有效说明路由消息互传的状态变化情况,以图 1 为例进行具体的分析。

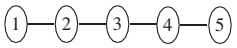

图 1 五个路由初始位置

五个路由初始化时,1-2-3-4-5,1 与 2、2 与 3、3 与 4、4 与 5 直接消息互传,且本路由知道本路由信息,则 $a_{1,2} = a_{2,3} = a_{3,4} = a_{4,5} = a_{1,1} = a_{2,2} = ,\cdots, = a_{5,5} = 1$,其余元素记为 0,如图 2(a)所示。最优禁位排列时,1-3-5-2-4,1 与 3、3 与 5、5 与 2、2 与 4 直接消息互传,置 $a_{1,3} = a_{3,5} = a_{5,2} = a_{2,4} = 1$,其余元素不变,如图 2(b)所示。

从图 2(b)可以看出,第 2 行(列)、第 3 行(列)元素全为 1,即路由 2 与路由 3 完成了所有信息的交互,再以后这两个路由开始脱落。当前只剩余路由 1、4、5 没有完成消息互传。那么开始深度优先遍历,第一次深度优先搜索,1-4-5-2-3,则置 $a_{1,4} = 1$,其余元素不变,如图 2(c)所示。

从图 2(c)可以看出,第 4 行(列)元素全为 1,即路由 4 此

时也完成了所有信息的交互,以后该路由脱落。当前只剩下路由 1 与 5 未完成消息互传。第二次深度优先搜索,1-5-2-3-4,则置 $a_{1,5}=1$,其余元素不变,如图 2(d)所示。此时表中元素全部为 1,说明五个路由完成了消息的互传。

ID	1	2	3	4	5
1	1	1	0	0	0
2	1	1	1	0	0
3	0	1	1	1	0
4	0	0	1	1	1
5	0	0	0	1	1

(a)

ID	1	2	3	4	5
1	1	1	1	0	0
2	1	1	1	1	1
3	1	1	1	1	1
4	0	1	1	1	1
5	0	1	1	1	1

(b)

ID	1	2	3	4	5
1	1	1	1	1	0
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	0	1	1	1	1

(c)

ID	1	2	3	4	5
1	1	1	1	1	1
2	1	1	1	1	1
3	1	1	1	1	1
4	1	1	1	1	1
5	1	1	1	1	1

(d)

图 2　图 1 中各路由消息互传状态分析

2.2　算法描述

Initially: $a_{i,i}=1, 1\leqslant i\leqslant n$

$a_{i,i+1}=a_{i+1,i}=1, 1\leqslant i\leqslant n-1$

Otherwise: $a_{i,j}=0, 1\leqslant i,j\leqslant n$

1 $P:=\{x_1,x_2,\cdots,x_n\}$ //optimal limit arrangement

$a_{x_k,x_{k+1}}=a_{x_{k+1},x_k}=1, 3\leqslant k\leqslant n-1$

2 while ($a_{i,j}\neq 1$ exists in the table)

To begin depth-first traversal

……

$r-th$ traversal

To traverse set τ of routing number

and then

modify these elements in the table

if(some routing number τ_i are off)

$r+1-th$ traversal

To traverse set $\tau=\tau/\tau_i$ of routing number

Randomly put τ_i behind the traversal result

else(no any routing number is off)

$r+1-th$ traversal

To traverse set τ of routing number

……

3 while (the value of all elements is 1)

end

3　实验分析

3.1　实验仿真

为了得到单链状态下的深度优先遍历次数,本文在 Intel Core2 Q8400 2.66 GHz CPU、4.00 GB 内存的 Windows 操作系统、VC6.0 环境下进行了 20 组实验。路由的个数基本均匀分布在 20 ~ 600 之间。实验结果如表 2 所示。

表 2　单链状态下路由个数与深度优先遍历次数

组号	路由	遍历	组号	路由	遍历	组号	路由	遍历
1	20	17	8	300	297	15	480	477
2	40	37	9	320	317	16	500	497
3	80	77	10	340	337	17	520	517
4	100	97	11	380	377	18	560	557
5	200	197	12	400	397	19	580	577
6	240	237	13	440	437	20	600	597
7	280	277	14	460	457			

由表 2 的内容,利用 Microsoft Excel 软件绘制出如图 3 所示的实验图。从图 3 看出,当路由个数分布在 20 ~ 600 之间时,算法中的深度优先遍历次数也随之增加,并且是线性增加。

3.2　算法演示

针对本文算法,采用图 4 进行算法的演示。

在图 4 中可以找到一个通路,不妨设为 A-D-E-B-C-F-G-H,将这八个路由依次编号为 1-2-3-4-5-6-7-8,即初始路由编号为{1,2,3,4,5,6,7,8}。根据算法进行演示:

a)最优禁位排列:{1,3,5,7,2,4,6,8}。

b)一次遍历后:{1,4,7,3,6,2,5,8}。

c)二次遍历后:{1,5,6,7,8,2,3,4},路由编号 5 和 2 开始脱落,即 D 和 C 已完成所有消息传递。

d)三次遍历后:{1,6,7,8,3,4,5,2},路由编号 6 和 3 开始脱落,即 E 和 F 已完成所有消息传递。

e)四次遍历后:{1,7,8,4,5,2,6,3},路由编号 7 和 4 开始脱落,即 B 和 G 已完成所有消息传递。

f)五次遍历后:{1,8,5,2,6,3,7,4},表中元素均为 1,所有路由都完成消息传递。

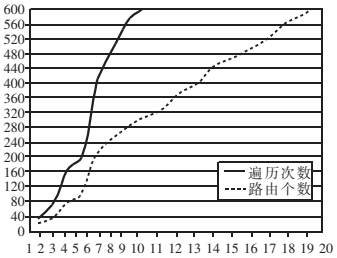

图 3　表 2 中数据的图示反映

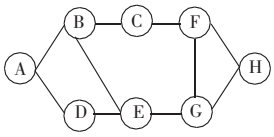

图 4　八个路由算法演示实验图

图 4 中所有路由传递信息流程如图 5 所示。

下面进行验证:任取路由,这里不妨取 B,其初始与 E、C 直接交互;最优禁位排列后,与 D、F 直接交互;一次遍历后,与 A、G 直接交互;二次遍历后,与 E 直接交互;三次遍历后,与 C、D 直接交互;四次遍历后,与 H、C 直接交互;五次遍历后,与 G 直接交互。在整个过程中,B 与 A、C、D、E、F、G、H 都进行过直接交互。同理,其他路由也都能得到验证。

3.3　算法时间复杂度

本文算法的时间复杂度与深度优先遍历的次数和每一次深度优先遍历的时间复杂度有关。在每一次深度优先遍历搜索过程中,本文采用的是类似二维数组的存储方式,所以深度优先遍历的时间复杂度是 $O(n^2)$。图 2 表明,在单链状态下,深度优先遍历次数的时间复杂度是 $O(n)$。那么结合起来,整个算法的时间复杂度就是 $O(n^3)$。传统的 Dijkstra 算法采用的是贪婪算法,即把所有的路由进行消息互传,其时间复杂度是 $O(n^3)$。从这一层面上看,本文的算法在单链状态下与 Dijkstra 算法具有相同的时间复杂度。然而实际的网络结构,其路由的排列并不仅仅是一个单链,它是一个由多个单链组成的多链复杂结构。针对这样的实际网络环境,文本的算法有明显的优越性。下面证明在实际的网络拓扑结构中,存在算法的时间复杂度在 $O(n^2)$ 到 $O(n^3)$ 之间或者该算法在时间复杂度为 $O(n^3)$ 时具有良好的优越性。

证明　当网络中路由处于单链状态下,如图 6(a)所示;当网络路由拓扑处于一种极好的状态下,即完全图形式,如图 6(b)所示。

图 5　信息传递流程

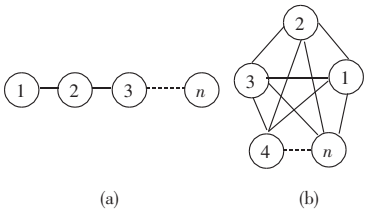

图 6　网络拓扑的两种极限状态

a)当网络结构处于单链状态下,从上文中第一段的分析论述可以看出,此时算法的时间复杂度是 $O(n^3)$,也就是该算法此时的适应程度最差。

b)当网络中的路由排列处于完全图时,两两路由能够进行一次性的直接交互,即同一时间内完成了信息的传递,此时深度优先遍历次数的时间复杂度为 $O(1)$,那么算法的总体时间复杂度是 $O(n^2)$。

c)通过 a)b)看出,算法的整体时间复杂度要么存在于 $O(n^2)$ 与 $O(n^3)$ 之间,要么仍处于 $O(n^3)$。然而对于后者,每增加一个环路,算法的遍历次数有所下降,并且在算法执行的过程中,大多数路由都是成对脱落。基于这两点因素,即便算法的深度优先遍历次数的时间复杂度仍处于 $O(n)$,但很大程度上提高了收敛速度,因此同样具有良好的优越性。

事实上,网络中设备的变动、增加减少,或者线路逻辑上、物理上的断开,都会造成网络动荡,从而改变了网络的拓扑结构。这种情况下,路由需要重新计算,以保证达到路由的一致性。路由重新计算,即执行本文算法,在深度优先搜索过程中,随着遍历次数的增加,某些路由编号开始脱落,这样,后面就不必花费时间遍历这些路由了,很大程度上缩减了整个深度优先搜索的时间。以图 1 为例,最优禁位排列之后,就出现了路由 2 和 3 脱落的现象,那么在执行第一次深度优先搜索时,只需对路由 1、4 和 5,而不再需要对全部的路由进行算法。第一次优先搜索之后,路由 4 脱落,那么在执行第二次深度优先搜索时,只需对路由 1 和 5 执行算法。如果不引用路由脱落办法,整个深度优先搜索过程中,要对 1 号到 5 号路由遍历 2 次,共 10 次;引用路由脱落策略,第一次深度优先搜索遍历 1 号、4 号、5 号路由,第二次深度优先搜索遍历 1 号、5 号路由,仅需要 5 次,相比 10 次而言,大大提高了算法的收敛速度,这在实际复杂多变的网络结构中具有较强的适应性。

3.4 本文算法较之 Dijkstra 算法的主要优越性

Dijkstra 算法是基于路径代价的一种算法,而本文算法是基于消息直接互传的一种算法。本文算法抛开了路径代价,省略了路径之间的代价加和,引入了禁位排列,将路由虚拟编号,进行多次禁位排列,在每次排列当中,路由都是直接交互的,从而简化了算法的形式。

4 结束语

路由决策是当前网络消息互传的关键技术,一个好的路由决策算法能够很快地使网络消息达到一致性。本文从数学原理中的禁位排列问题出发,将排列数看做网络拓扑中的路由。路由在网络中进行信息交换,等同于排列数在各个禁位排列之后的不断变化。这种基于禁位排列的路由决策算法能够满足网络的基本需求,较快地使网络链路状态达到一致。

然而,由于本文算法是一个全新的算法,因此有些问题一时半会尚未得到解决,如兼并考虑模式 1 和 2 两种禁位排列时,其满足条件的排列数是多少。笔者查阅了大量的文献资料,并没有现成的理论支持,因此只是通过编程实现而并未给出相关的数学证明。另外,虽然从理论上说明了算法的时间复杂度存在于 $O(n^2)$ 和 $O(n^3)$ 之间,但毕竟对多链状态的仿真实验做得不够充分。在以后的工作中,笔者将沿着这两个方向继续探究,找到某个阈值范围内的路由排列链路数,使得该算法的时间复杂度低于 Dijkstra 算法的时间复杂度。

参考文献:

[1] 谢希仁. 计算机网络[M]. 5 版. 北京:电子工业出版社,2008:144-156.

[2] 彭中君,魏永林. 基于无线自组网的多接口路由协议[J]. 计算机工程与应用,2010,46(19):113-116.

[3] 徐佳,王汝传. 基于效用的容迟网络路由技术研究[J]. 计算机研究与发展,2011,28(4):1211-1215.

[4] 岐世峰,李艳华,梅大成. 蚁群算法在 QoS 单播路由中的应用研究[J]. 电子科技大学学报,2010,39(2):271-274.

[5] 胥小波,郑康锋. 基于并行 BP 神经网络实现的路由查找算法[J]. 通信学报,2012,33(2):61-67.

[6] 孙侠,殷志祥. 全错位排列问题的基于芯片的 DNA 计算模型[J]. 大学数学,2010,26(5):79-81.

[7] 李三燕,徐允庆. 二重错排与拉丁矩的计数[J]. 宁波大学学报:理工版,2011,24(2):51-55.

[8] GOBJAKA H, BREITHART Y. Discovering network topology of large multisubnet Ethernet networks[C]//Proc of IEEE INFOCOM. 2007: 428-434.

[9] 李琼琳. 容斥原理及应用[J]. 中国科技信息,2012,20(8):58-59.

[10] GOLDA A F, ARIDHA S, ELAKKIYA D. Algorithmic agent for effective mobile robot navigation in an unknown environment[C]//Proc of International Conference on Intelligent Agent & Multi-Agent Systems. 2009:1-14.

[11] 张德富,彭煜,张丽丽. 求解三维装箱问题的多层启发式搜索算法[J]. 计算机学报,2012,35(12):2553-2560.

(上接第 256 页)

[8] WANG Jiong, MEDIDI S. Topology control for reliable sensor-to-sink data transport in sensor networks[C]//Proc of IEEE International Conference on Communications. Piscataway: IEEE Press, 2008: 3215-3219.

[9] 徐强,汪芸. 容错节能无线 WSN 中可靠覆盖问题的解决方案[J]. 软件学报,2006,17(11):184-191.

[10] 李小龙,林亚平,易叶青,等. 传感器网络中基于虚拟坐标的节点调度方案[J]. 软件学报,2008,19(8):2089-2101.

[11] YIN Zhen-yu, ZHAO Hai, LIN Kai, *et al*. A coverage-preserving node scheduling scheme based on trust selection model in wireless sensor networks[C]//Proc of the 1st International Symposium on Pervasive Computing and Applications. Piscataway: IEEE Press, 2006: 696-698.

[12] GANERIWAL S, SRIVASTAVA M B. Reputation-based framework for high integrity sensor networks[C]//Proc of the 2nd ACM Workshop on Security of Ad hoc and Sensor Network. New York: ACM Press, 2004: 66-77.

[13] SHAIKH R A, JAMEEL H, D'AURIOL B J, *et al*. Group-based trust management scheme for clustered wireless sensor networks[J]. IEEE Trans on Parallel and Distributed Systems, 2009, 20(11): 1698-1712.

[14] HO J W. Zone-based trust management in sensor networks[C]//Proc of IEEE International Conference on Pervasive Computing and Communications. Piscataway: IEEE Press, 2009: 1-2.

[15] TIAN D, GEORGANAS N D. A node scheduling scheme for energy conservation in large wireless sensor networks[J]. Wireless Communications and Mobile Computing, 2003, 3(2): 271-290.