

[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 1/20 (2006.01)



[12] 发明专利申请公开说明书

[21] 申请号 200510129180.4

[43] 公开日 2006 年 8 月 16 日

[11] 公开号 CN 1819507A

[22] 申请日 2005.10.13

[21] 申请号 200510129180.4

[30] 优先权

[32] 2004.10.13 [33] US [31] 10/962,488

[71] 申请人 阿尔卡特公司

地址 法国巴黎

[72] 发明人 P·拉比诺维奇 B·麦克布莱德

[74] 专利代理机构 北京市中咨律师事务所

代理人 杨晓光 李 峰

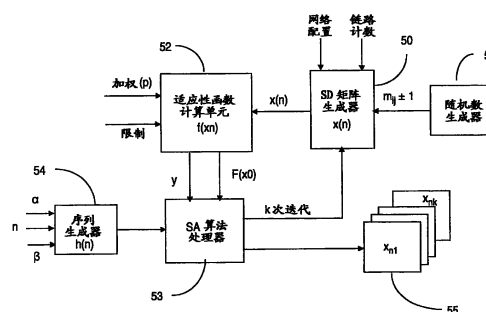
权利要求书 3 页 说明书 15 页 附图 6 页

[54] 发明名称

用于业务量矩阵估算的模拟退火

[57] 摘要

使用模拟退火算法, SATME 方法和系统估算源端到目的端业务量矩阵, 该业务量矩阵估算被表示为满足一系列给定限制的所有可能的矩阵集合上的概率分布。所述限制直接将用户知道的网络业务量信息作为目标函数(适应性函数)的分量进行编码, 然后使用模拟退火将该函数最小化。采用根据本发明的方法, 能够包括任何形式的任意限制, 而且在没有可行的解的情况下, 可以由目标函数进行诊断而不会收敛为零。



1. 一种用于基于给定的链路计数对通信网络估算业务量矩阵的方法，包括以下步骤：

a) 对相应网络建立适应性函数；

b) 生成初始的源端到目的端 (SD) 业务量矩阵并对所述初始 SD 业务量矩阵计算所述适应性函数的初始值；

c) 修改所述初始矩阵以获得随机修正 SD 业务量矩阵并为所述随机修正 SD 业务量矩阵计算所述适应性函数的当前值；

d) 选择对应于所述初始值和所述当前值中较小值的那个 SD 业务量矩阵作为临时 SD 业务量矩阵；

e) 使用模拟退火算法改变所述临时 SD 业务量矩阵，直到所述临时 SD 业务量矩阵的所述适应性函数达到最小值；并且

f) 选择在步骤 e) 中估算的所述临时 SD 业务量矩阵作为可能的 SD 业务量矩阵并在预先选择的次数内重复步骤 b) 到 e)。

2. 根据权利要求 1 的方法，其中步骤 a) 包括：

为所述适应性函数确定表示实际网络配置和链路计数的绝对限制，并指定硬性能损失 (hard penalty) 给各个绝对限制；

基于业务量模型对所述适应性函数建立软限制，并指定软性能损失 (soft penalty) 给各个软限制；

将对相应链路的各个限制转化为所述适应性函数的元素；并

作为加权元素的和计算所述适应性函数，各个加权表示软限制或硬限制。

3. 根据权利要求 2 的方法，其中所述软性能损失小于所述硬性能损失。

4. 根据权利要求 2 的方法，其中对于网络各个链路的所述绝对限制包括所述相应链路的所述链路计数。

5. 根据权利要求 2 的方法，其中所述软限制包括指定的业务量模型。

6. 根据权利要求 4 的方法，其中使用距离函数计算所述适应性函数的

各个元素。

7. 根据权利要求6的方法，其中根据所述网络中使用的路由选择协议和所述相应的链路计数，将适应性函数的各个元素作为一个或多个观察值之间的平方差来计算，所述观察值说明了使用所述相应链路的所有路径。

8. 根据权利要求1的方法，其中步骤b)中的所述初始SD业务量矩阵的所述元素均匀地选在最小值：

$$(x_{ij}) = \min\{\text{节点 } N_i \text{ 发出的总业务量, 到达节点 } N_j \text{ 的总业务量}\}。$$

9. 根据权利要求8所述方法，其中步骤c)包括：

均匀选择所述初始业务量矩阵的元素；

以预先选定的整数随机增大或减小所述元素的值。

10. 根据权利要求9的方法，其中所述预先选定的整数是1。

11. 根据权利要求1的方法，其中步骤e)包括

在0和1之间产生均匀随机数“r”；

将所述临时SD业务量矩阵的所述元素确定为：

$$x_n = \begin{cases} y & r < \min\{1, e^{(f(y)-f(x_{n-1}))/h(n)}\} \\ x_{n-1} & \text{否则} \end{cases}$$

其中 $h(n)$ 是定义用于对所述适应性函数进行最小化的迭代次数 n 的序列函数。

12. 根据权利要求10的方法，其中 $h(n)$ 是几何衰减函数 $h(n) = \beta \alpha^n$ 。

13. 一种用于对通信网络估算业务量矩阵的系统，包括：

源端到目的端(SD)业务量矩阵生成器，用于基于网络配置数据和对所述网络的各个链路收集的链路计数生成初始SD业务量矩阵；

随机数生成器，用于随机地改变所述初始SD业务量矩阵以获得随机修正SD业务量矩阵；

适应性函数计算单元，用于对所述相应的SD业务量矩阵确定初始适应性函数和更新的适应性函数并选择对应于所述初始值和当前值中较小值

的那个 SD 业务量矩阵作为临时 SD 业务量矩阵；以及

模拟退火（SA）算法处理器，用于改变所述临时 SD 业务量矩阵，直到所述临时 SD 业务量矩阵的所述适应性函数达到最小值，并选择对应于最小值的所述 SD 业务量矩阵作为可能的 SD 业务量矩阵。

14. 根据权利要求 13 的系统，其中所述模拟退火算法处理器使用序列生成器用于确定迭代次数以获得所述最小值。

15. 根据权利要求 13 的系统，其中所述适应性函数计算单元基于限制计算所述适应性函数。

16. 根据权利要求 13 的系统，其中所述适应性函数表示为加权元素之和，各个元素转化对相应链路的限制。

用于业务量矩阵估算的模拟退火

技术领域

本发明涉及估算通信网络的性能，特别是用于业务量矩阵估算（SATME）的模拟退火方法和系统。

背景技术

由于科技进步的刺激，电信产业在过去十年经历了迅猛发展。台式计算机处理能力的进步以及应用程序响应时间的缩短促进了相应的网络基础结构的改进。需要网络业务增长支持的新的带宽消耗应用和设备应用的迅速改变使网络供应商面临是否网络有足够的容量支持目前和将来的业务需求的问题。

在任何业务量和/或网络工程任务中，对网络了解更多只会有利。特定的应用要求知道：例如，从A站点到B站点有多少业务量、特定的用户从A站点发送多少业务量到B站点、网络中的过热点在哪里或者需要增加多少容量以及在哪里增加。在特定条件下的关于网络运转情况的信息也是有用的。例如，网络供应商可以基于对下列情况的了解来优化网络性能：如果链路出现故障必须转移多少业务量、如果两点间的业务量从链路被转移到另一链路会发生什么、是否大多数发源于子网络（自治系统、国家或公司）的业务量停留在子网络中，等等。这些消息通常用于路由选择通路计算、负载平衡等。

网络工程工具被设计用于处理由业务量需求的增加或现有需求的改变而引起的这些问题并确定对网络的必要扩充和调整以支持规划的业务量。这些工具通常有助于修改网络设计和路由方案来避免业务量瓶颈、实现服务的差别计价等。然而，所有这些工具要求了解业务量统计数字，尤其是

实时的源端-目的端的业务量需求。

针对基于数据包的网络保持这些统计数字的方法就是查看各个数据包并记录其源端和目的端。然而，这种方法需要巨大的表格来记录这些信息。而且，如果需要的是针对网络子集的源端到目的端业务量矩阵，则在没有查看源端和目的端的地址并确定数据包要采用的由路由选择信息指示路径的情况下，子集的入口和出口都是不可用的。这对于巨大的网络和大量的数据包是不可行的。

将业务量统计数字组成“业务量矩阵”也是已知的。业务量矩阵通常被表示为数据的总字节或数据包的总量或所有节点的源端-目的端对之间的连接。虽然当前的路由器能够测量在各个接口（输入和输出链路）的输入和输出业务量的计数，但是精确无误地从链路计数推导出业务量矩阵通常是不可能的。能做到的最多就是从观测的链路业务量作出关于业务量矩阵的概率推断。

美国电话电报公司提出通过组合两个已知的模型，即重力模型与层析（tomographic）模型，从链路负载的有效测量来推断业务量矩阵。重力模型假设：对于输出业务量，站点之间的业务量与每个对等基础（相对于每个路由器基础）上的各个站点的业务量是成比例的，而且不同位置上的业务量之间没有系统性差异。层析模型基于模型应用链路限制。这些模型的组合减小了大型网络的问题规模而且提供了比独立使用的重力模型和层析模型更精确的结果。

然而，层析-重力（tomo-gravity）方法不考虑随机限制的简单指定，例如已知链路的速度是另一个链路的两倍，则其可能具有大约两倍的业务量。而且，该方法基于业务量遵循“重力”模型这样的假设，而实践中不能这样假设。

以上描述的现有技术，诸如监测所有的数据包、重力、层析方法和层析-重力，分别地，都不能满足大的业务量而且依赖于脱离现实的假设。另一方面，有必要为现代通信中的业务量工程提供工具和技术，特别是用于估算业务量模型（特性），例如业务量、过热点、故障影响等。

发明内容

本发明的目的是为业务量工程提供工具和技术，完全或部分的减少现有工具和技术缺陷。

具体地，本发明的目的是使用模拟退火算法估算源端到目的端的业务量矩阵，该业务量矩阵估算被表示为概率分布。

因此，本发明提供了基于测量的链路计数为通信网络估算业务量矩阵的方法，包括以下步骤：a) 为相应的网络建立适应性函数；b) 生成初始源端到目的端（SD）业务量矩阵并对初始 SD 业务量矩阵计算适应性函数的初始值；c) 修改初始矩阵以获得随机修正 SD 业务量矩阵并对随机修正 SD 业务量矩阵计算适应性函数的更新值；d) 选择对应于初始值和更新值中较小值的那个 SD 业务量矩阵作为临时 SD 业务量矩阵；e) 使用模拟退火算法改变临时 SD 业务量矩阵，直到临时 SD 业务量矩阵的适应性函数达到最小值；并且 f) 选择在步骤 e) 中估算的临时 SD 业务量矩阵作为可能的（potential）SD 业务量矩阵并在预先选择的次数内重复步骤 b) 到 e)。

本发明还提供了用于为通信网络估算业务量矩阵的系统。该系统包括：源端到目的端（SD）业务量矩阵生成器，用于基于网络配置数据和对网络的各个链路收集的链路计数生成初始 SD 业务量矩阵；随机数生成器，用于随机地修改初始 SD 业务量矩阵以获得随机修正 SD 业务量矩阵；适应性函数计算单元，用于为所述相应的 SD 业务量矩阵确定初始适应性函数和更新的适应性函数并且选择对应于初始值和当前值中较小值那个 SD 业务量矩阵作为临时 SD 业务量矩阵；以及模拟退火（SA）算法处理器，用于改变临时 SD 业务量矩阵直到临时 SD 业务量矩阵的适应性函数达到最小值，并且选择对应于该最小值的 SD 业务量矩阵作为可能的 SD 业务量矩阵。

有利地，本发明不需要关于业务量的假设，也就是说，其允许用户以简单的方式指定关于网络和业务流的先验信息。

而且，本发明不仅提供了各个元素的点估算，更确切地说，提供了涵盖所有可能的概率分布，所以能够估算可能的变化和不同链路上的业务量

之间的相关性。

附图说明

本发明上述和其它的目的、特征和优点，将通过下面的优选实施例更详细的说明变得清楚，如附图所示，其中：

图 1 是用于说明本发明的网络的简单例子；

图 2 是示出当前在图 1 的网络上建立的连接的例子的源端到目的端（SD）矩阵；

图 3a 是关于图 1 中具有图 2 所示的源端到目的端连接的网络配置的链路业务量表；

图 3b 针对图 2 的 SD 矩阵，直观地示出了各个连接在图 1 网络的各个链路上所沿的路径；

图 3c 针对图 2 的 SD 矩阵，说明关于图 1 的网络的链路连接表；

图 4 示出了关于图 1 的网络的 SD 矩阵，其中连接数要确定的；

图 5 说明根据本发明用于业务量矩阵估算（SATME）系统的模拟退火的框图；

图 6a 示出了使用本发明的方法、通过实验获得的关于图 4 的 SD 矩阵的分布举例；

图 6b 是图 4 矩阵的两个元素间的联合分布举例。

具体实施方式

模拟退火是开发用于冶金处理的优化技术，在退火的时候通过使用控制参数来模拟烤炉的温度。因此称作温度（T）的该控制参数，被设置初始值而且根据预定的进度表随时间的改变而降低。

基础模拟退火被成功的使用在 VLSI 结构设计的例子中，为了更好地理解该算法这里将其以普通术语解释。通过移动元件或交换两个元件的位置就改变了初始配置；随机改变该布局以得到新布局。控制参数 T 表明接受导致增加成本的移动（搬移）的可能性。搬移之后，评估新的布局成本

而且如果成本降低了, 该配置就会被接受。否则, 新的配置取决于 T 的概率被接受。然后, 使用例如指数函数来减小 T , 而且当 T 减小到很低的值时该过程被停止, 这个很低的值要求只有减小成本的移动才被接受。该算法以关于 T 的很高的值开始, 该值逐渐减小, 使得增加成本的搬移被接受的可能性逐渐降低。这样, 该算法达到最佳或接近于最佳的配置。

该技术也被应用在蜂窝网络中的频率分配。然而, 将这个机制应用到通信网络中并非简单的任务。异构网络的严重问题是模拟退火算法受到在搜索中无数的局部最小量的限制。一旦该算法陷入搜索空间的无价值的最小量, 随机避免该最小值的机会就很低了。

本发明的主要思想是着眼于产生表示为概率分布的业务量矩阵估算, 使模拟退火算法适应通信网络。该解决方案的优点在于其不需要关于业务量的假设 (即, 其允许用户用简单的方式指定关于网络和业务量的先验信息), 而且估算矩阵能够估算所有可能的变化以及元素之间的相关性。

根据本发明的 SATME (用于业务量矩阵估算的模拟退火) 方法使用与模拟退火不同的方式。

主要区别在于模拟退火算法确定目标函数的 (接近于) 最优解, 然而业务量矩阵估算需要的是在满足一组给定限制的所有可能的表格设置上的均匀分布。需要在所有可能性上的均匀分布的原因是为了避免偏离可能的答案。因为有很多将满足给定链路计数的可能的业务量矩阵, 通过直接应用 SA 算法所提供的单个答案是不准确的。理想地, 该方法应该提供最可能的业务量矩阵以及一些其可变性的测量。从本发明提供的信息来计算两者都是可能的。

增加到 SA 算法的另一个改进是限制的使用。根据本发明, 用户所知道的网络业务量信息直接作为限制被编码。该限制作为目标函数 (能量) 的分量被编码, 接着使用模拟退火将该目标函数最小化。采用根据本发明的方法, 可以包括任何形式的任意限制, 而且在没有可行的解的情况下可以由目标函数进行诊断而不会收敛为零。

图 1 示出了用于说明本发明的网络 100 的一个简单例子。如图所示,

该网络有五个通过链路 10、12、14、16、18 及 20 连接的节点 N1 - N5。更具体地，由于业务量是双向的，下标与确认某个链路的附图标记相关以反映方向。因此，对于节点 N1 和 N2 之间的链路， 10_{12} 用于表示从节点 N1 到 N2 的方向， 10_{21} 用于从节点 N2 到 N1 的方向。这里使用的术语“连接”用于端到端（源端到目的端）业务量而且在这里是指使用字母 C 来后面跟随连接节点的数字。例如，节点 N1 到 N5 的连接用 C15 表示。

连接可以沿着多条路径被选择路由。例如，网络 100 可以将发自节点 N1 并且去往节点 N4（即，连接 C14）的业务量的路由选择为沿着由链路 10_{21} 、 12_{23} 、 14_{34} 组成的路径，或沿着只由链路 16_{14} 组成的路径。对连接的路径（也被称为端到端路由）选择是由网络使用的路由选择算法产生的。由于大多数网络采用最短路径算法，上面的例子中 N1 - N4 业务量将选择沿着链路 16_{14} 的路由。

图 2 是连接表，也称为源端到目的端（SD）表（或矩阵），示出了图 1 的网络 100 上的节点 N1 到 N5 之间当前建立的所有连接的例子。可以看到，连接是双向的；例如从节点 N1 到节点 N5 的连接（C15）数量是 4，从节点 N5 到节点 N1 的连接（C51）数量是 5。由相应节点发出的连接（出局连接）的总数在最后一列中给出，由相应节点接收的连接（入局连接）的总数在最后一行中给出。在这个例子中，节点 N1 发出 19 个连接并终止 20 个连接。

图 3a 是链路业务量表，示出了假设网络使用最短路径路由选择算法，网络 100 如何选择这些连接的路由（沿着哪些特定的链路）。例如图 2 所给出的，链路 10_{12} 具有 13 个连接，即 5 个 C12 连接（N1 和 N2 之间）、4 个 C13 连接（N1 和 N3 之间）和 4 个 C15 连接（N1 和 N5 之间）。作为另一个例子，链路 20_{35} 具有 23 个连接，即 4 个 C15 连接（N1 和 N5 之间）、7 个 C25 连接（N2 和 N5 之间）、5 个 C35 连接（N3 和 N5 之间）和 7 个 C45 连接（N4 和 N5 之间）。

链路业务量表的最后一列提供了在一个方向上沿着链路的连接总数，包括迂回连接以及在相应的方向由节点发出的连接。最后一行提供了进入

节点的连接总数，包括该节点在另一个方向上的连接和转发到其它节点的连接。这些数目在这里被称为“链路计数”。

网络 100 的拓扑结构（见图 1）以及最短路径选择意味着业务量沿着如图 3b 所示路径传送。例如，连接 C51 被从节点 N5 到节点 N3（沿着链路 20₅₃）、从节点 N3 到节点 N2（沿着链路 12₃₂）和从节点 N2 到节点 N1（沿着链路 12₂₁）的路径所用。

“倒转”这个信息，提供了沿各个链路上传送了哪些业务量，如图 3c 所示。该表格实际上也给出了使用相应的链路的源端-目的端对（连接）。

如上所述，在现代网路中链路计数信息是现有的；然而，源端到目的端信息不是现有的。根据本发明，通过使 SA 算法适应链路业务量矩阵问题，链路计数被用来建立源端到目的端业务量矩阵。也就是说，根据本发明的方法充分利用了 SA 的性能来探索限制空间的领域。

因为有很多可能的源端到目的端业务量矩阵能产生如图 3a 所示的链路计数，一些可能性分布的特征可以用来限制解的数量。尤其是，本发明使用许多限制作为性能损失（penalty），然后寻求最小化性能损失总量的解。此外，满足这些限制的解有很多，但是要寻找的本质是在所有这些可能性之上的均匀分布。

图 4 示出了对于图 1 的 5 节点网络的 SD 矩阵，其中连接数是要确定的；它们由“m”后面跟着代表相应源端-目的端节点对的下标来表示。例如，m₅₄表示节点 N5 和 N4 之间的连接数。该思想是以特定的方式使用 SA 算法并使用表示问题的不同部分的限制来确定 SD 矩阵（原始 SD 矩阵）的元素。

限制被分为两类，即绝对限制和软限制。硬限制是最重要的限制，说明在各个入口点和出口点所观察的实际业务量（数据包和字节的数量），换言之，链路业务量观测值的链路计数。其它硬限制表示路由选择规则，例如最短路径路由选择或任何预先指定的路径。软限制可以表示例如网络的先验知识，例如特定链路典型地提供其它链路业务量两倍的服务。

为了阐明根据本发明 SA 的应用如何工作，结合图 3a 的链路业务量表，

再次使用图 1 的网络例子。对于任何网络的第一绝对限制是没有任何连接可以在同一个节点发起和终止。对于这个特定的例子（5 个节点），这个绝对限制用下式表示：

$$m_{11}=m_{22}=m_{33}=m_{44}=m_{55}=0$$

下面的绝对限制是图 3a 所提供的例子中由每个链路的总连接数所产生的。

- 节点 N1 发出的业务量总数是 19。
- 节点 N2 发出的业务量总数是 39。
- 节点 N3 发出的业务量总数是 60。
- 节点 N4 发出的业务量总数是 22。
- 节点 N5 发出的业务量总数是 21。
- 到达节点 N1 的业务量总数是 20。
- 到达节点 N2 的业务量总数是 38。
- 到达节点 N3 的业务量总数是 51。
- 到达节点 N4 的业务量总数是 29。
- 到达节点 N5 的业务量总数是 23。

以上的绝对限制必须满足，而且违反这些限制会导致严重的性能损失。可以在该列表里加入许多“较软的”限制，即有较小性能损失的限制。例如我们说在链路 10_{12} 上的业务量应该与链路 14_{43} 上的业务量相同。

$$m_{12}=m_{43}$$

结合绝对限制，这里给出了下列可以用于确定 SD 矩阵的元素的等式。

A.绝对限制：

$$m_{11}=m_{22}=m_{33}=m_{44}=m_{55}=0$$

$$m_{12}+m_{13}+m_{15}+m_{14}=19$$

$$m_{21}+m_{31}+m_{51}+m_{13}+m_{15}+m_{23}+m_{25}+m_{24}=39$$

$$m_{31}+m_{32}+m_{51}+m_{52}+m_{34}+m_{54}+m_{15}+m_{25}+m_{35}+m_{45}=60$$

$$m_{41}+m_{42}+m_{43}+m_{45}=22$$

$$m_{51}+m_{52}+m_{53}+m_{54}=21$$

$$m_{21}+m_{31}+m_{51}+m_{41}=20$$

$$m_{12}+m_{13}+m_{15}+m_{31}+m_{32}+m_{51}+m_{52}+m_{42}=38$$

$$m_{13}+m_{15}+m_{23}+m_{25}+m_{43}+m_{45}+m_{51}+m_{52}+m_{53}+m_{54}=51$$

$$m_{14}+m_{24}+m_{34}+m_{54}=29$$

$$m_{15}+m_{25}+m_{35}+m_{45}=23$$

B.关于以上例子的软限制是:

$$m_{12}+m_{13}+m_{15}=m_{43}+m_{45}$$

然后, 加权 (或性能损失) 被指定到这些限制的违例上。大的性能损失被指定到上面 A 下所列的限制, 比如说 $p_1=100$; 而较小的性能损失被指定到 B 上的软限制, 比如说 $p_2=25$ 。

限制用于产生适应性函数 $f(m)$, 然后使用 SA 算法将该函数最小化。本发明提出作为加权平方项的和而构成的目标函数。绝对限制作为对于相应方向的已知链路总数 (从链路业务量表) 与当前估算的链路总数 (估算矩阵的相应的元素 m 之和) 之间的差被引入。按照它们所指定的权值, 软限制也被引入。因此, 适应性函数变成:

$$\begin{aligned} f(m) = & p_1(m_{12}+m_{13}+m_{15}+m_{14}-19)^2 + \\ & p_1(m_{21}+m_{31}+m_{51}+m_{13}+m_{15}+m_{23}+m_{25}+m_{24}-39)^2 + \\ & p_1(m_{31}+m_{32}+m_{51}+m_{52}+m_{34}+m_{54}+m_{15}+m_{25}+m_{35}+m_{45}-60)^2 + \\ & p_1(m_{41}+m_{42}+m_{43}+m_{45}-22)^2 + \\ & p_1(m_{51}+m_{52}+m_{53}+m_{54}-21)^2 + \\ & p_1(m_{21}+m_{31}+m_{51}+m_{41}-20)^2 + \\ & p_1(m_{12}+m_{13}+m_{15}+m_{31}+m_{32}+m_{51}+m_{52}+m_{42}-38)^2 + \\ & p_1(m_{13}+m_{15}+m_{23}+m_{25}+m_{43}+m_{45}+m_{51}+m_{52}+m_{53}+m_{54}-51)^2 + \\ & p_1(m_{14}+m_{24}+m_{34}+m_{54}-29)^2 + \\ & p_1(m_{15}+m_{25}+m_{35}+m_{45}-23)^2 + \\ & p_1(m_{11}^2+m_{22}^2+m_{33}^2+m_{44}^2+m_{55}^2) + \end{aligned}$$

$$p_2(m_{12}+m_{13}+m_{15}-m_{43}-m_{45})^2$$

然后，选择初始 SD 业务量矩阵，用 x_0 表示。该初始矩阵的元素 (x_{ij}) 均匀地选在最小值：

$$(x_{ij}) = \min\{\text{节点 } N_i \text{ 发出的总业务量, 到达节点 } N_j \text{ 的总业务量}\}$$

然后计算关于这个被提议矩阵的适应性函数，在这里用 $f(x_0)$ 表示。现在，均匀选择这个矩阵的元素，即，使用标准的随机数生成器，以相同概率选择各个元素。该元素然后被随机地增大或减小。例如，如果在抛出硬币时，该硬币正面朝上，则所选元素加“1”，否则减“1”。

这一变化产生随机修正 SD 业务量矩阵 y ，关于该矩阵的适应性函数 $f(y)$ 被再次计算。

如果 $f(y) < f(x_0)$ 则设置 $x_1: y$

如果 $f(y) > f(x_0)$ 则在 0 和 1 之间生成均匀随机数 “ r ”

$$y \quad r < \min\{1, e^{(f(y)-f(x_{n-1}))/h(n)}\}$$

$$x_n = \begin{matrix} \\ \text{否则} \\ x_{n-1} \end{matrix}$$

这个过程被重复许多次而且当适应性函数接近于 0 时，意味着系统已经达到稳定状态。在各个连续迭代中得到的中间矩阵称为临时 SDF 业务量矩阵。对应于稳定状态的临时矩阵 x_n 作为满足所有限制的可能的 SD 业务量矩阵被接受。

上述操作被重复许多次，每次都记录得到的矩阵作为可能的答案。对于相应的有关性能损失，这些可能的矩阵集合是满足限制的所有可能矩阵组的近似均匀样本。

函数 $h(n)$ 被定义为序列函数，其依赖迭代次数 n 。以下形式的几何衰减函数是优选使用的，不过其它函数同样是可取的（例如对数函数）。当几何衰减函数被使用时：

$$h(n) = \beta \alpha^n$$

$$\beta = \frac{1}{2}$$

$$\alpha = 0.9995$$

常数 α 和 β 的选择影响 SATME（适应性函数与迭代次数）的性能但是如上选择通常在实践中使用。

图 5 说明用于生成可能的 SD 矩阵组 55 的本发明的 SATME 系统。SATME 系统包括 SD 矩阵生成器 50、随机数生成器 51、适应性函数计算单元 52、SA 算法处理器 53 和序列生成器 54。SD 矩阵生成器 50 基于已知链路计数和已知网络配置（节点数和节点间的链路），生成具有最小值上均匀选出的元素的初始矩阵 x_0 。适应性函数计算单元 52 为呈现给它的矩阵计算适应性函数。在这种情况下，该单元为初始矩阵计算初始适应性函数 $f(x_0)$ ；如上所述，使用加权的绝对和软限制计算 $f(m)$ ，而且 $f(m)$ 被呈现给处理器 53 以进行最小化。

随机数生成器 51 随机选择初始矩阵中的一个元素并且再随机地增大/减小该元素。在初始矩阵中，所选元素的新值取代旧值，产生随机修正 SD 矩阵 y 。适应性函数生成器 52 现在为随机修正 SD 矩阵计算新的当前适应性函数 y ，再将其提供给处理器 53。

SA 算法处理器 53 比较初始和当前适应性函数并开始根据序列生成器 54 提供的序列，使用迭代和随机数 r 对当前函数的值进行最小化处理。序列生成器 54 对 SA 算法使用所选的函数类型。如上所述，对生成器 54，几何衰减函数是被优先选择的，而且函数的参数是用户可配置的。迭代次数 n 也是用户可配置的。当适应性函数接近于 0 时，系统达到稳定状态，而且对可能的 SD 矩阵 x_n 使用相应的矩阵元素 m 。该操作重复许多次，每次记录得到的矩阵作为可能的答案。

现在我们回到前面用过的例子中的网络 100，假设初始矩阵 x_0 如下。

36	64	18	74	50
21	99	54	65	45
97	24	68	31	15
71	73	85	51	33
90	96	76	2	0

在 50,000 次迭代之后，适应性函数变得接近于 0 而且系统达到稳定

状态。图 6a 和 6b 示出了在 100 次迭代以后可以得到的 SD 业务量矩阵的两个元素间的相关性如何。这些图形目的在于发现 m_{12} 和 m_{34} 之间的相关性。例如，在这些图中， m_{12} 的范围是 0 到 17，因为在分析中，可以从直方图中看出的 m_{12} 最大值是 17。因此， m_{12} 首先被箝位在 0 而且对于 m_{12} 的这个值的 m_{34} 的条件分布被确定。然后， m_{12} 变为 1，等等，直到 $m_{12}=17$ 。

因为相关性是线性测量，这可以有利于查看两个元素 m_{12} 和 m_{34} 之间的联合分布，如图 6b 所示。

m_3/m_{12}	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	3	1	0	1	3	1	1	3	1	0	3	0	0	1	0	0	1	0
1	1	2	3	3	0	2	3	2	5	2	0	0	1	3	1	0	2	1
2	5	3	2	5	0	5	4	2	1	1	6	1	4	0	2	1	1	1
3	4	2	5	8	7	2	5	4	4	2	4	1	3	4	1	1	0	0
4	7	6	9	3	3	6	5	1	5	2	5	3	3	3	0	4	1	1
5	5	4	5	4	4	5	7	4	4	0	4	4	2	4	6	3	3	3
6	4	3	5	9	5	4	4	8	6	6	3	3	4	1	2	7	5	3
7	10	8	7	4	4	4	9	7	7	6	3	7	4	5	4	1	4	2
8	4	5	8	8	4	8	8	12	7	7	4	5	1	7	1	6	3	4
9	9	8	5	4	12	6	6	7	5	5	3	2	2	5	6	4	3	8
10	8	11	7	8	3	3	13	3	3	7	10	6	5	9	4	3	5	6
11	6	7	8	7	7	7	6	6	9	10	4	5	10	7	6	9	7	9
12	3	6	5	5	11	9	5	4	5	3	6	14	10	10	6	9	9	4
13	5	12	11	5	7	9	2	8	9	7	8	12	4	6	12	7	6	10
14	6	6	4	8	3	10	4	5	2	8	6	6	6	5	8	7	8	3
15	5	5	9	7	8	3	5	4	5	6	5	3	9	11	10	6	2	10
16	5	3	0	4	5	4	4	4	5	2	7	4	8	3	11	11	8	7
17	4	1	3	2	7	5	3	3	8	7	5	6	9	2	7	6	7	8
18	3	4	1	2	4	4	2	4	3	3	5	6	4	4	5	3	10	5
19	2	1	1	0	2	2	3	5	4	8	4	3	2	4	3	6	5	9
20	0	1	2	1	1	0	1	1	2	4	1	5	3	2	3	3	4	2
21	1	1	0	2	0	1	0	2	0	2	3	2	5	1	1	1	5	1
22	0	0	0	0	0	0	0	1	0	2	1	1	1	3	1	2	0	2

总的说来，本发明通过先在较大的表格空间均匀地选择初始的解，来

改进已知的退火算法。初始解的各个元素在 $[0, m]$ 间均匀选择，其中 m 是对应于该元素的行和列总和的最小值。注意，得到的表格将（除非很幸运）不满足任何限制（除了非负性）。

第二个改进将所有的限制作为目标函数分量进行编码，该函数采用当前解的观察值和目标值（限制）之间的差的加权平方和的形式。可以选择该权值以反映先前关于限制的重要性的看法 - 因此，行和列的总和有很高的权值，而其它限制可以具有较小的权值。采用这套方法，可以包括任何形式的任意限制，而且在没有可行的解的情况下，可以由目标函数进行诊断而不会收敛为零。

附录

实施该 SATME 方法的算法的代码如下:

SA[tt_, its_, temp0_, alpha_, plot_] :=

Module[{i, pivot, coin, testMatrix, te, t, p}

t=tt;

p = Table[0, {its}];

For[i = 1, i <= its, i++,

tempi = temp0*alpha^i;

pivot = {Random[Integer, {1,5}],Random[Integer, {1,5}]};

coin = If[Random[] > 0.5, 1,-1];

testMatrix = t;

te = testMatrix[[pivot[[1]], pivot[[2]]] += coin;

If[te < 0, te = 0,];

testMatrix[[pivot[[1]], pivot[[2]]] = te;

If[fitness[testMatrix] < fitness[t], t = testMatrix,

If[Random[] < Min[1, Exp[(fitness[testMatrix] – fitness[t])/tempi]]

t = testMatrix,];

p[[i]] = fitness[t];

];

Return[t]

]

fitness[t_] := Module[

{ans, pl, p2, cl, c2, c3, c4, c5, c6, c7, c8, c9, c10, c11, c12},

pl = 100;

p2 = 25;

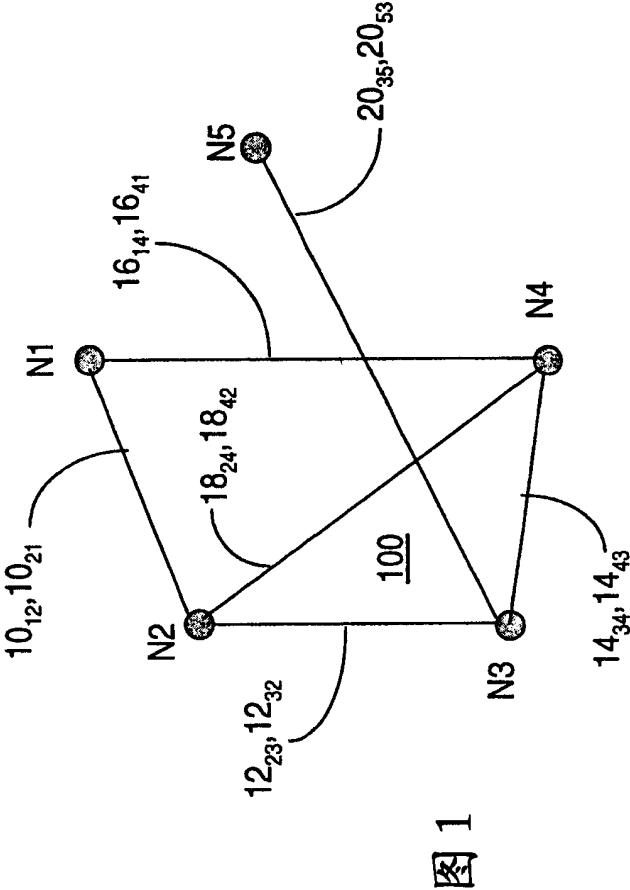
c1 = (t[[1, 2]] + t[[1,3]] + t[[1,5]] + t[[1, 4]]- 19) ^2;

c2 = (t[[2, 1]] + t[[3, 1]] + t[[5, 1]] + t[[1, 3]] + t[[1, 5]] +

```

    t[[2, 3]] + t[[2, 5]] + t[[2, 4]] - 39)^2;
c3 = (t[[3, 1]] + t[[3, 2]] + t[[5, 1]] + t[[5, 2]] + t[[3, 4]] +
    t[[5, 4]] + t[[1,5]] + t[[2, 5]] + t[[3, 5]] + t[[4, 5]]-
    60)^2;
c4 = (t[[4, 1]] + t[[4, 2]] + t[[4, 3]] + t[[4, 5]] -22)^2;
c5 = (t[[5, 1]] + t[[5, 2]] + t[[5, 3]] + t[[5, 4]] - 21)^2;
c6 = (t[[2, 1]] + t[[3, 1]] + t[[5, 1]] + t[[4, 1]] -20)^2;
c7 = (t[[1,2]] + t[[1,3]] + t[[1,5]] + t[[3, 1]] + t[[3, 2]] +
    t[[5, 1]] + t[[5, 2]] + t[[4, 2]] - 38)^2;
c8 = (t[[1,3]] + t[[1, 5]] + t[[2, 3]] + t[[2, 5]] + t[[4, 3]] +
    t[[4, 5]] + t[[5, 1]] + t[[5, 2]] + t[[5, 3]] + t[[5, 4]] -
    51)^2;
c9 = (t[[1,4]] + t[[2, 4]] + t[[3, 4]] + t[[5, 4]] -29)^2;
c10 = (t[[1,5]] + t[[2, 5]] + t[[3, 5]] + t[[4, 5]] - 23)^2;
c11 = (t[[1,2]] + t[[1,3]] + t[[1,5]] - t[[4, 3]] - t[[4, 5]])^2
c12 = (t[[1, 1]]^2 + t[[2, 2]]^2 + t[[3, 3]]^2 + t[[4, 4]]^2 +
    t[[5, 5]]^2);
ans=
    p1*(c1 + c2 + c3 + c4 + c5 + c6 + c7 + c8 + c9 + c10) + p2*c11
    p1*c12;
Return[ans]]

```



连接表 (源端至目的端)

自 / 至	N1	N2	N3	N4	N5	总数
N1	0	5	4	6	4	19
N2	5	0	2	7	7	21
N3	5	8	0	7	5	25
N4	5	4	6	0	7	22
N5	5	3	4	9	0	21
总数	20	20	16	29	23	108

图 2

图 3a

链路业务量表

自 / 至	N1	N2	N3	N4	N5	总数
N1	0	13	0	6	0	19
N2	15	0	17	7	0	39
N3	0	21	0	16	23	60
N4	5	4	13	0	0	22
N5	0	0	21	0	0	21
总数	20	38	51	29	23	161

图 3b

业务量路径

自 / 至	N1	N2	N3	N4	N5
N1	0	1→2	1→2→3	1→4	1→2→3→5
N2	2→1	0	2→3	2→4	2→3→5
N3	3→2→1	3→2	0	3→4	3→5
N4	4→1	4→2	4→3	0	4→3→5
N5	5→3→2→1	5→3→2	5→3	5→3→2	0

链路 - 连接表

链路	连接
10 ₁₂	C12, C13, C15
16 ₁₄	C14
10 ₂₁	C21, C31, C51
12 ₂₃	C13, C15, C23, C25
18 ₂₄	C24
12 ₃₂	C31, C32, C51, C52
14 ₃₄	C34, C54
20 ₃₅	C15, C25, C35, C45
16 ₄₁	C41
18 ₄₂	C42
14 ₄₃	C43, C45
20 ₅₃	C51, C52, C53, C54

图 3c

连接表 (源端至目的端)

自 / 至	N1	N2	N3	N4	N5
N1	m ₁₁	m ₁₂	m ₁₃	m ₁₄	m ₁₅
N2	m ₂₁	m ₂₂	m ₂₃	m ₂₄	m ₂₅
N3	m ₃₁	m ₃₂	m ₃₃	m ₃₄	m ₃₅
N4	m ₄₁	m ₄₂	m ₄₃	m ₄₄	m ₄₅
N5	m ₅₁	m ₅₂	m ₅₃	m ₅₄	m ₅₅

图 4

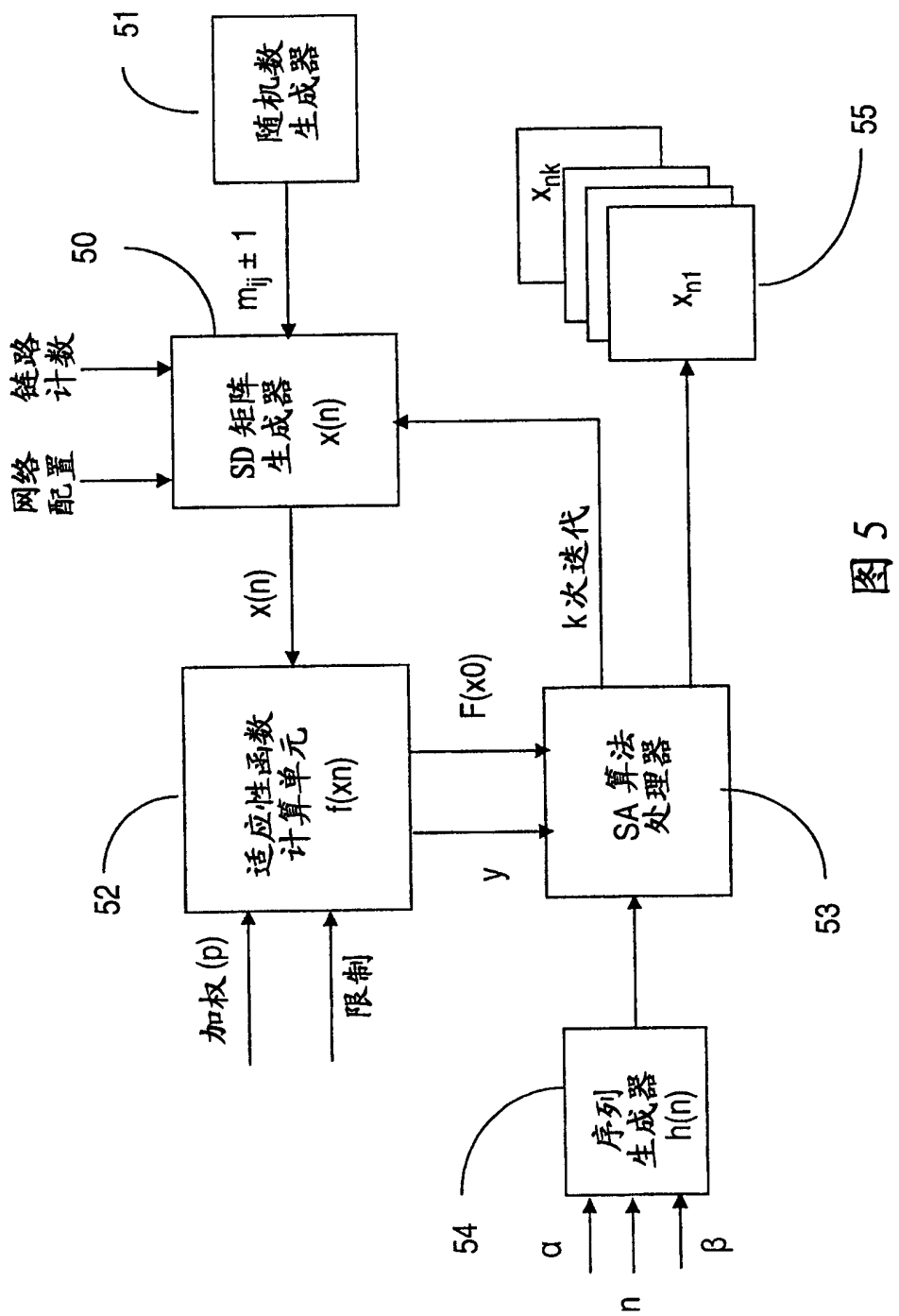


图 5

图 6a

