学校代码：10286
分类号：TP311
密　级：公开
U D C：004.6
学　号：131440

东南大学
SOUTHEAST UNIVERSITY

硕士学位论文

基于流量及可靠性的不确定图边关键度研究

研究生姓名：**李富豪**

导 师 姓 名：**张柏礼 副教授**

申请学位类别 **工 学 硕 士**　学位授予单位 **东 南 大 学**

一级学科名称 **计算机科学与技术**　论文答辩日期 **2016 年 06 月 03 日**

二级学科名称　学位授予日期 **20　年　月　日**

答辩委员会主席 **陆宁云**　评　阅　人 **汪鹏**

院盲

2016 年　月　日

東南大學

硕士学位论文

基于流量及可靠性的不确定图边关键度研究

专 业 名 称： 计算机科学与技术

研究生姓名： 李富豪

导 师 姓 名： 张柏礼

RESEARCH ON THE KEY EDGE OF UNCERTAIN GRAPH BASED ON FLOW AND RELIABILITY

A Dissertation Submitted to

Southeast University

For the Academic Degree of Master of Engineering

BY

Li Fu-hao

Supervised by

Associate Professor Zhang Bai-li

School of Computer Science and Engineering

Southeast University

June 2016

东 南 大 学 学 位 论 文 独 创 性 声 明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得东南大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

研究生签名：_____日　期：_____

东 南 大 学 学 位 论 文 使 用 授 权 声 明

东南大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权东南大学研究生院办理。

研究生签名：_____导师签名：____日　期：_____

摘　要

不确定图的关键边是指相对于其他边而言，能够在更大程度上影响网络结构与功能的一些特殊边，一旦发生故障，将对整个系统产生巨大影响，甚至导致系统瘫痪。目前，网络中关键边挖掘因其广泛的应用价值和理论研究意义，受到众多研究人员的关注，各种针对特定应用需求的边关键度评估方案不断被提出。这些方法站在各自的立场，从不同角度为我们探索不同应用需求下节点的重要性提供可选的方案。本文提出适用于流网络的关键边评价模型，针对网络中边损毁（故障）后对流量和可靠性产生的相对损失来综合衡量关键边，重点在于如何在动态变化的网络环境中对最大流和可靠性进行快速计算，从而实现动态流网络中边关键度既准确又快速的评估。论文主要包括以下几个方面：

（1）从边损毁（故障）后对流量和容量可靠性产生的相对损失这一角度，构建了一种基于流量和容量可靠性的关键边衡量模型，对边的关键度进行综合评估。针对此模型，首先设计了一种基于最大流和容量可靠性重新计算的原始算法 BASE_FCP，然后针对其性能存在的不足，提出基于流量和容量可靠性的状态区间缓存算法 SCA_FCP，其首先将不确定图的边定性分类，然后根据不同的边设计不同的计算方法以减少复杂度。最后，实验分析表明，SCA_FCP 相对于 BASE_FCP 虽然在空间复杂度有一定的增加，但在时间复杂度方面具有较大的优势。

（2）虽然基于流量和容量可靠性的模型对于关键边的区分度较好，但不确定图容量可靠性的计算过于复杂，而计算不确定图分布可靠性只需要获取其极小子图即可，为此，本文重新构建了一种基于流量和分布可靠性的关键边模型。针对此模型，首先提出了基于流量和分布可靠性的状态区间缓存算法 SCA_FDP，但是考虑到其在状态划分过程中存在初始状态划分空间大的缺点，为此，提出一种基于确定边过滤的优化算法 SCA_FEA，该算法使用割集边和悬挂边对于初始划分集合剪枝，从而减少初始状态划分集合大小。实验表明基于流量和分布可靠性的模型比基于流量和容量可靠性的模型，不管在时间还是空间性能上都有提升，相对于算法 SCA_FDP，SCA_FEA 也使得时间性能有了一定程度的提升。

（3）针对基于流量和分布可靠性的关键边模型，使用算法 SCA_FEA 计算不确定图（最可靠最大流）分布可靠性依然是一个 NP-Hard 问题，难以满足对计算速度要求很高的实时环境中，为此，提出一种基于流量和分布可靠性的关键边近似算法 KEAA_FDP，该算法利用最大流分布的剩余图，通过流量调整，选取最可靠简单路径进行增广，实现（最可靠最大流）分布可靠性快速计算，并给出了近似算法获取的关键边序列解和精确算法解的相似性度量。实验表明算法 KEAA_FDP 相对于 SCA_FEA 算法性能有了显著提升，能够适应不同规模和稠密度的图，且波动性不大。

关键词: 不确定图，关键边，最大流，容量可靠性，分布可靠性

Abstract

The key edge of uncertain graph can affect the network structure and function, once destroyed (or be removed), the entire system will have a great influence, and even cause the system to crash. At present, the key edge mining in the network has been concerned by many researchers because of its extensive application value and theoretical research significance. These methods stand in their respective positions, from different angles to explore the key edge of different application needs. This paper proposed an key edge evaluation model of flow network, which according to the relative loss of the flow and reliability after the edge are removed (or destroyed) to measure the key edge. We focus on how to calculation the maximum flow and reliability of dynamically network after removing edges fastly. The paper mainly includes the following aspects:

(1) Construct a measurement model based on the relative loss of the flow and capacity reliability after the edge is removed. As for this model, we designed a kind of based algorithm BASE_FCP and aiming at the shortage of its performance we proposed algorithm SCA_FCP, which first classify the edge then use different method to calculation for different kind edges. Finally, the experimental results show that SCA_FCP has a certain increase in space complexity compared to BASE_FCP, but it has a great advantage in time complexity.

(2) Although the flow and capacity reliability model has a good distinction, but the capacity reliability calculation is too complicated, however the calculation of distribution reliability only need to get the minimal graph of uncertain graph. Therefore, we construct a new model based on the relative loss of the flow and distribution reliability after the edge is removed. As for the model, we proposed algorithm SCA_FDP, but considering the shortcomings of the initial state space partition in the division of state, we proposed optimization algorithm SCA_FEA based on the determined edge filtering which can reduce the size of initial state section. Experimental results show that the flow and distribution reliability model has advantage of time and space performance to the flow and capacity reliability model. SCA_FEA has a certain advantage in time complexity compared to SCA_FDP

(3) As for the flow and distribution reliability model, to calculate the most reliable maximum flow distribution reliability of uncertain graph is still an NP-hard problem, so it is difficult to meet the computational speed in a real-time environment. Therefore, we propose an approximation algorithms KEAA_FDP. We Gives the similarity measure of the key sequence solutions that get by approximate algorithm and exact algorithms. Then we analyze the worst situation. The experimental results show that KEAA_FDP has a significant improvement on the performance compared with SCA_FEA, and it can adapt

to different size and density of the graph, and the volatility is not very large.

Key words: uncertain graph, key edge, maximum flow, distribution reliability, capacity reliability

目录

第 1 章 绪论

1.1 问题研究背景与意义

关键边是指相对于其他边而言，能够在更大程度上影响网络结构与功能的一些特殊边[1]。这里的网络结构包括度分布、平均距离、连通性、聚类系数、度相关性等，网络功能涉及网络的最大流、可靠性、抗毁性、传播、同步、控制等[2]。关键边一般数量非常少，但其影响却可以快速地波及到网络中大部分节点和边 [3]。例如，在对一个无标度网络的蓄意攻击中，少量最重要节点被攻击就会导致整个网络瓦解[4-5]；微博中最有影响力的几个用户所发的微博很快就能传遍整个网络[6]。为此，针对不确定图边的重要度进行评估、寻找关键边就成为一项有意义的工作，这对于后期网络维护具有重大的决策和指导意义。如俄亥俄州克利夫兰市的几条烧断的高压线能够造成北美大停电事故，导致数百亿美元的损失。如果我们能够事先对这个电力网络的结构有所了解，并找到关键线路，采取预防措施，就可能避免如此巨额的经济损失[2]。这里最核心的问题就是如何识别这些重要的线路。

目前，各种针对特定应用需求的边关键度评估方案不断被提出，这些方法站在各自的立场，从不同角度为我们探索不同应用需求下节点的重要性提供可选的方案。本文提出适用于流网络的关键边评价模型，针对网络中边移除（故障）后对流量和可靠性产生的相对损失来综合衡量关键边，重点在于如何在动态变化的网络环境中, 对最大流和可靠性进行快速计算, 从而实现动态流网络中边关键度既准确又快速的评估。

1.2 研究现状

1.2.1 网络关键节点相关研究

目前，研究人员已经提出非常多的重要节点挖掘算法，这些方法站在各自的立场，从不同的角度为我们探索不同背景下的重要性提供科学方案，纵观各种方法，它们基本上遵从以下几个思路[1]：

（1）从节点的局部环境考虑。节点的局部环境包括直接邻居、间接邻居及连边、节点的聚类系数等。ClusterRank[7]同时考虑节点的度和聚类系数。度中心性考虑节点直接邻居的数量，半局部中心性考虑节点四层邻居的信息。在考虑数量的同时，还有一些挖掘方法从邻居节点的重要性相互增强角度进行了探索，这主要是指基于特征向量的系列方法。另外，接近中心性[8]、Katz 中心性[9]和信息指标从节点与全局范围内所有节点的联系强弱角度评估节点的重要性。

（2）从节点所处的位置考虑。这里包括节点在路径中的位置和节点在网络中的位置两方面。前者主要包括图中心性[10]、介数中心性[11]及其他基于路径的挖掘方法，而 k-壳分解法则[12]是后者的典型代表。

（3）从节点对网络功能的影响考虑。这类方法主要考察将节点移除后网络结构和功能的变化，例如，节点删除的最短距离法和残余接近中心性主要关注网络中平均最短距离的变化[13]，生成树法关注节点删除后网络生成树的变化[14]，节点收缩法关注节点删除后网络凝聚度的变化[15]。

对于上述的（1）基于节点局部环境和（2）基于节点所处的位置考虑的模型，存在“简单而不准确”的问题，而对于（3）从节点对网络功能的影响考虑的模型又面临“准确但太复杂”的挑战。因此，良好的边关键度评估模型是非常难得的，往往需要深入地结合应用，充分利用各种约束设计针对性求解算法才有可能获得。

本文所提的关键边评估模型属于其中的第（3）类。针对不确定流网络中边移除（故障）后对流量和可靠性产生的相对损失，来综合评估关键边。该模型将流量作为衡量关键边的最重要因素，当流量一致时，比较最大流容量可靠性。重点在于如何在动态变化的网络环境中，对最大流和可靠性进行快速计算，从而实现动态流网络中边关键度既准确又快速的评估。

1.2.2 其他相关研究

不确定图是一种特殊的边动态变化的网络，本文是对于不确定图的边关键度的研究，针对不确定图中边移除后对最大流和可靠性（容量可靠性或分布可靠性）产生的相对损失，来评估关键边。本节首先研究了最大流、容量可靠性和不确定图流分布可靠性的相关研究。

最大流问题是网络流理论的重要组成部分，它是一类经典的组合优化问题，同时也可以看作是特殊的线性规划问题，被广泛的应用在众多的领域。

针对这个问题的研究有 40 多年的历史[16]。典型的最大流算法包括：网络单纯形法[17]、Ford-Fulkerson 方法[18]、Edmonds-Karp 方法[19]和 Dinic 方法[17,20]等，当前的最大流算法主要可以分为两大类：增载算法[20-22]和预流推进算法[23-25]。

一般情况下最大流通常对应多个不同的分布方案，对于确定图而言，这些不同的分布方案对于流的传递是等价的，但是对于边、节点都可能存在不确定性的不确定图来说，不同的流分布方案对应着不同的可靠性，从而存在着最可靠最大流问题。例如：在输变电网络中，电能从发电站 A 点传输到目的地 B 点可能存在着多种传输方案，而这些方案由于涉及到具体不同可靠性的输电设备和输电线路而具有不同的可靠性，为了保证电网电能传递具有更好的安全可靠性，就需要找到最可靠的一种传输电能的方案，即不确定图的最可靠流分布。

不确定图流分布的可靠性是网络流问题在不确定图上的自然延伸，它是一类重要的最优决策问题，关系到如何最大限度的利用现有网络条件选择最可靠的最大流传输方案，对解决实际中诸如构建可靠性网络、选取可靠传输路线以及分析系统薄弱环节等一系列问题有重要的研究意义[26]。目前不确定图流分布的可靠性的计算方法主要是基于简单路径组合分流的最可靠最大流算法 SPCA 和基于极小子图的最可靠最大流算法 MSBA，这两类算法分别从不同的角度有效地解决不确定图最可靠最大流问题[27-28]。

对于随机流网络容量可靠性的计算，可以分为两大类：精确方法和近似方法。精确方法中完全枚举算法是最基本的算法，它通过直接枚举网络的所有可能状态得到网络的可靠度。Chin-Chia Jane 与 Yih-Wenn Laih 采用了基于 d-flows[25]和基于 d-cuts[29]的不同状态空间划分方法，将状态空间划分成能够满足流量阈值需求的合格状态集，不能够满足流量阈值需求的不合格状态集和同时包含了合格状态和不合格状态的待划分状态集，通过迭代的方式进一步处理待划分状态集，最终通过计算合格状态集中所有合格状态的概率之和来得到网络可靠性。一些研究人员还提出了基于 d-flows 或 d-cuts 的算法对状态空间进行过滤[30-33]，然后运用容斥原理来计算网络可靠性。

对于复杂的系统，精确算法的运行时间代价往往过高而让人难以容忍，为了从准确性和算法执行效率中找到一种折中的方案，实际系统中通常会选取近似算法来代替精确算法。为此，许多近似算法被相继提出，Ramirez-Marquez 和 Coit[34]提出了蒙特卡洛模拟算法，Rocco 和 Muselli[35]则采用机器学习方法成功的获得网络可靠性。

Lin[32,36-37]还将网络可靠性问题扩展到节点不可靠的情况，采用最小路径集的方法进行了研究。考虑到网络中的每条边及节点都有一定的传输代价参数，Lin and Yeh 研究了在实际应用中基于传输代价限定的网络可靠性问题[31-32]，这其实是一类具有多条件约束的网络可靠性问题。对于一个拓扑结构给定的网络，如何从一堆给定的多状态元件中选取该网络需求数量的元件来构建一个实际的网络，同时又需要满足该网络所要满足的传输流量、代价、时间、可靠性等诸多因素限制的实际需求，也是一类比较经典的网络元件分配与网络可靠性结合的研究问题。针对这类问题，Yi-Kuei Lin 和 Cheng-Ta Yeh 提出了相应的遗传算法[38]。

1.3 研究内容

本文首先提出一种适合流网络的关键边评价模型，针对不确定图中边移除（故障）后对流量和可靠性产生的相对损失，来衡量关键边，本文将重点研究不确定图边移除之后，如何对于剩余不确定图的最大流和可靠性进行快速计算。本文主要研究的内容主要包括以下几个方面：

（1）网络关键边评估方案研究。网络中关键环节挖掘的研究已经持续多年，各种针对特定应用需求的挖掘算法被不断提出，本文则针对不确定动态流网络，分别提出基于流量和容量可靠性的关键边评估模型以及基于流量和分布可靠性的关键边评估模型。

（2）网络关键边评估精确方法的研究。针对基于流量和分布可靠性的关键边评估模型，本文提出对应的状态区间缓存算法，将不确定图的边首先定性分类，当需要计算可靠性（容量可靠性和分布可靠性）的时，首先判断边的类边，然后根据不同的子算法获取边损毁之后动态网路的可靠性。

（3）网络关键边近似算法的研究。针对基于流量和分布可靠性的关键边模型，计算不确定图（最可靠最大流）分布可靠性依然是一个 NP-Hard 问题，难以满足对计算速度要求很高的实时环境中，本文提出一种基于流量和分布可靠性的关键边近似算法

KEAA_FDP，实现（最可靠最大流）流分布可靠性的快速计算，并给出了近似算法产生的关键边序列解和精确算法解的相似性度量。

1.4 本文内容组织

本文的内容组织如下：

第 1 章 绪论。本章论述了不确定图等相关问题的研究背景，以及不确定图中关键边识别的意义，并调研了该问题的研究现状，简要介绍了本文研究的主要内容。

第 2 章 不确定图及关键边相关理论。本章主要介绍相关的理论，首先介绍了最大流的相关概念和常用的算法，然后介绍了随机流网络可靠性的基本概念，接着介绍了不确定图的相关概念、不确定图数据的一般建模方法、不确定图分布可靠性以及容量可靠性等相关问题，最后介绍了目前常用关键节点算法分析。

第 3 章 基于流量和容量可靠性的关键边衡量方法。本章从边移除（故障）后对流量和容量可靠性产生的相对损失这一角度，构建了一种基于流量和容量可靠性的关键边衡量模型，对边的关键度进行综合评估。针对此模型，首先设计了一种基于最大流和容量可靠性重新计算的原始算法 BASE_FCP，然后针对其性能存在的不足，提出了基于流量和容量可靠性的状态区间缓存算法 SCA_FCP。

第 4 章 基于流量和分布可靠性的关键边衡量方法。本章重新构建了一种基于流量和分布可靠性的关键边模型。针对此模型，首先提出了基于流量和分布可靠性的状态区间缓存算法 SCA_FDP，但是考虑到其在状态划分过程中存在初始状态划分空间大的缺点，提出了一种基于确定边过滤的优化算法 SCA_FEA。

第 5 章基于流量和分布可靠性的关键边近似算法。本章提出一种基于流量和分布可靠性的关键边近似算法 KEAA_FDP，实现（最可靠最大流）流分布可靠性的快速计算，并给出了近似算法产生的关键边序列解和精确算法解的相似性度量。

第 6 章 总结和展望。本章总结了本文所做的主要工作，并对下一步的研究进行展望。

第 2 章 不确定图及关键边相关理论

本文是基于网络最大流、随机流网络可靠性和不确定图最可靠最大流分布可靠性对不确定图关键边进行的相关研究。而本章主要介绍与之相关的理论，首先介绍了最大流的相关概念和常用的算法，然后介绍了随机流网络可靠性的基本概念，接着介绍了不确定图的相关概念、不确定图数据的一般建模方法、不确定图分布可靠性以及容量可靠性等相关问题，最后介绍了目前常用网络关键节点的算法。

2.1 最大流问题

2.1.1 基本概念和相关定义

最大流问题（maximum flow problem）一种组合最优化问题。网络流理论研究的一个基本问题是求网络中一个可行流 f，使其流量达到最大，这种流 f 称为最大流，这个问题称为（网络）最大流问题。最大流问题是一个特殊的线性规划问题，是图论的一个重要分支。一般用标号法寻求最大流比用求线性规划问题的一般方法要方便得多。为了系统的描述最大流问题，本节首先给出了最大流的基本概念和相关定义。

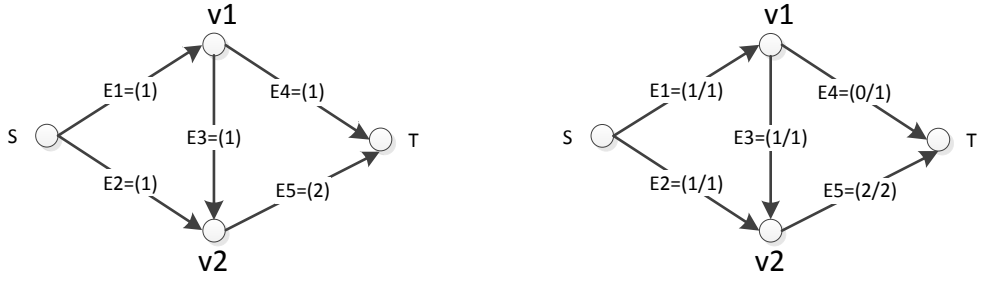
定义 2-1.[39]（流网络） 一个流网络 $G(V,E)$是一个有向图，它满足以下两个特征：（1）对于任意的边 $e(u, v)\in E$，均有一个非负容量 $c(e)\geq 0$；（2）存在单一的源点 s 与单一的汇点 t，且 $s\in V, t\in V$。其中，V 是 G 的顶点集，E 是 G 的边集。

定义 2-2.（*s-t* 流） 在流网络 G 中，给定源点 s，汇点 t，*s-t* 的流是一个边集 E 到实数集 R 的映射 $f: E \to R$，且 f 满足如下两条性质：

（1） 容量约束：对所有的 $e\in E$ ，$f(e) \leq c(e)$；

（2） 流守恒性：对所有顶点 u 和 v ($u \in V$，$v \in V$-{s, t})，满足$\Sigma_{e\in\{(u, v)\,|\,(u, v)\,\in E\}} f(e) = \Sigma_{e\in\{(v, u)\,|\,(v, u)\,\in E\}} f(e)$，即流入顶点 v 的流量之和等于从顶点 v 流出的流量之和，对于源点 s 与汇点 t 满足$\Sigma_{e\in\{(s, u)\,|\,(s, u)\,\in E\}} f(e) - \Sigma_{e\in\{(u, s)\,|\,(u, s)\,\in E\}} f(e) = \Sigma_{e\in\{(u, t)\,|\,(u, t)\,\in E\}} f(e) - \Sigma_{e\in\{(t, u)\,|\,(t, u)\,\in E\}} f(e)$，表示从源点流出的净流量（$s$ 流出的所有流量与流入 s 的所有流量之差）最终都会流入汇点，并且该净流量为 *s-t* 流值。

定义 2-3.（最大流） 在流网络 G 中，给定源点 s，汇点 t，其中最大的 s-t 流值为 s 到 t 的最大流。



（a）流网络　　（b）*s-t* 流

图 2-1 流网络及 *s-t* 流

如图 2-1 所示，（a）为流网络，（b）为 *s-t* 流，源点 *s* 到汇点 *t* 的最大流为 2。

在实际应用中，许多最大流问题中的容量及流量通常都是整数，而对于容量及流量为有理数的情况，也可以通过适当的比例变换，将它们转换成整数。本文为了将问题一般化，所做的研究是在整数容量的基础下进行的。

2.1.2 常见的最大流算法

关于最大流问题的相关算法有很多，其中比较基础的算法是 Ford-Fulkerson[40]算法，该算法依赖三个重要的概念：剩余图，增广路径和割[39]，它们也是其它许多最大流算法的实现基础。本节将主要介绍一些常用的最大流算法及算法对应的时间复杂度。

（1）Ford-Fulkerson 算法

Ford-Fulkerson 是目前基本的解决最大流的方法，算法是一种迭代算法，它的核心思想是：在初始状态下，对所有的 $u, v \in V$，$f(u, v)=0$，即每条边上流量的初值等于 0，然后，在算法的每次迭代过程中，通过不断在 *f* 对应的“剩余图”G_f 上寻找一条“增广路径”*p* 来增加流值。其中“增广路径”可以简单的看成是从源点 *s* 出发到汇点 *t* 的一条路径，沿着这条路径可以将 *s* 的流量更多的压入到 *t*，从而增加流 *f* 的值。反复执行这个过程，直到 G_f 中不存在增广路径为止。下面给出的是 Ford-Fulkerson 算法的伪代码实现。

算法 2-1. 最大流算法 Ford-Fulkerson 算法

Input: G, souce s, sink t
output: the max flow of s-t

```
1. for each edge (u,v) ∈ E(G)
2.       f(u, v) ← 0
3.       f(v, u) ← 0
4. End
5. Gf  ← G
6. while there exists a path p from s to t in Gf
7.          cf(p) ← min{cf(u,v) where (u,v) ∈ p}
8.          for each edge (u,v) ∈ p
9.                   f(u, v) ← f(u,v) + cf(p)
10.                  f(v, u) ← - f(u,v)
11         End
12.         recomputed Gf for flow f
13. End
```

下面对 Ford-Fulkerson 算法相关的一些概念及理论进行说明。

定义 2-4.（剩余图） 对于给定的流网络 $G(V, E)$，源点为 *s*，汇点为 *t*，设 *f* 为 *G* 中 *s-t* 流，则 *f* 在 *G* 中对应的剩余图 $G_f(V', E')$定义如下：

(a) $V' = V$;

(b) 对顶点 $u, v \in V$，如果存在边 $e(u, v) \in E$，则存在 $e'(u, v) \in E'$， 且 e' 容量 $c(e') = c(e)–f(e)$，此时称该边为正向边；

(c) 如果 G 中边 $e(u, v)$上的流量 $f(e) \neq 0$，则在 G_f 中存在从 v 到 u 的边 $e(v, u)$，其容量 $c(e') = f(e)$，此时称该边为反向边。

如图 2-2，G_f 是图 2-1(b)中流 f 对应于 2-1(a)中流网络 G 的剩余图。

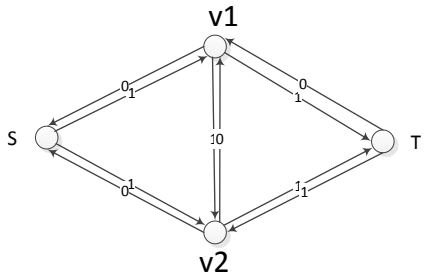


图 2-2 剩余图 G_f

在最大流问题的研究中，最大流最小割定理是最大流算法分析的理论基础，下面简单介绍一下割的概念及该定理。

定义 2-5.（*S*-*T* 割） 流网络 $G(V, E)$的割(S, T)是顶点集 V 的一个划分，它将 V 划分成 S 和 $T(T = V - S)$两个部分，且该划分满足源点 $s \in S$， $t \in T$，同时，将割的容量定义为从 S 到 T 的边容量之和，即 $c(S, T) = \Sigma_{e \in \{(u, v) \mid u \in S \wedge v \in T \wedge (u, v) \in E\}} c(e)$。

定理 2-1.[39]（最大流最小割定理）在流网络 $G(V, E)$中，f 是源点 s 到汇点 t 的一个流，则下面三个条件具有等价关系：

(a) f 是 G 的一个最大流。

(b) 剩余图 G_f 中不包含 s-t 增广路径。

(c) G 中存在某个割(S, T)满足$| f | = c(S,T)$。

最大流最小割定理不仅说明了 Ford-Fulkerson 算法的正确性，同时也说明了网络的最大流等于最小割。

Ford-Fulkerson 算法的运行时间与增广路径的选取密切相关，算法运行时间的上界为 $O(|E||f_{max}|)$[39]。如果$| f_{max} |$很大，而迭代过程每次选择的增广路径能够增加的流量比较小，则算法的效率会很低。

为了合理的选择增广路径，Edmonds 和 Karp[28]采用广度优先的搜索方式，在剩余图中，将每条边的距离看成单位 1，寻找 s 到 t 的最短路径作为增广路径来增加流。已经证明 Edmonds-Karp 算法的时间复杂度为 $O(|V||E|^2)$。

（2）Dinic 算法

Dinic[50]算法是基于分层网络和阻塞流思想的最大流算法，它将剩余图 G_f 中的每条边长度看成 1，在每次迭代过程中，根据 G_f 中的顶点与源点 s 的距离来构建 G_f 对应的分层网络 MSN，通过在 MSN 中寻找 s 到 t 的阻塞流对原始流 f 进行更新，直到汇点 t 不在当前的分层网络中时停止。下面给出的是 Dinic 算法的伪代码实现。

算法 2-2. 最大流算法-Dinic 算法

Input： G, souce s, sink t
Output： the max flow of s-t

1. **for** each edge $(u,v) \in E(G)$
2. $f(u,v) \leftarrow 0$
3. $f(v,u) \leftarrow 0$
4. **End**
5. $G_f \leftarrow G$
6. compute the *MSN* for G_f starting from source *s*
7. **while** sink *t* is in *MSN*
8. find a blocking flow f^* in *MSN*
9. **for** each edge (u,v) in *G*
10. $f(u,v) \leftarrow f(u,v) + f^*(u,v)$
11. **End**
12. compute G_f for flow *f*
13. recomputed *MSN* for G_f
14. **End**

Dinic 算法和 push-relabel[39]算法都是当前使用较为广泛的最大流算法，它们的时间复杂度均为 $O(|V|^2|E|)$。

2.2 随机流网络可靠性相关问题

2.2.1 基本概念和相关定义

网络的可靠度是网络性能的一个重要指标，可靠性分析一直是各研究者研究的热点问题，网络的复杂性、动态性、多态性等特点使得传统成熟的可靠性评估方法，如可靠性框图（RBD）法、故障树分析（FTA）法等很难用于网络。目前，对网络可靠性的研究取得了很多成果，网络可靠性的内涵也由传统的基于网络拓扑结构的连通可靠性逐渐拓展到考虑网络流的容量可靠性，并向基于业务等考虑用户需求的性能可靠性延伸。

网络可靠性评估主要分为 2 类[42]：（1）网络连通可靠性评估，指的是仅考虑网络拓扑结构，将“网络实现连通功能的概率”作为可靠性度量；(2)网络容量可靠性评估，它在考虑网络是否连通的基础上，还考虑了网络中链路和节点的容量，将“存在满足一定流量需求的连通路径的概率”作为可靠性度量；

根据网络或其组成部分的状态的多少网络可分为二态（binary state）网络和多态（multi-state）网络。

（1）二态（binary state）网络是指网络或网络的每个元素（节点或边只有两种状态：好/坏。网络的每条边的容量的取或为 0，或为某一个整数。

（2）有的网络的组成部分具有多种状态或容量。在某些况下，除了考虑网络状态的好与坏，还要研究不同的性能指标。这种网络我们称为多态网络，这样的网络又称为随机流网络。

现阶段为了简化问题，暂时先考虑二态情况，故本文中所使用的所有图和网络，在不特殊说明的情况下都是二态的。

定义 2-6.（随机流网络的容量可靠性）设 G=(N，A，M)是从源点 s 到汇点 t 的一个随机流网络，其中 N 为节点集，A={a_i | 1≤i≤n}为边集，M={M^1，M^2，…，M^n}，M^i 为边 a_i 的最大容量，i=1,2，…，n。那么 G 的 d 容量可靠性可以表示为$Pr = \sum_{g_1}^{g_n} p(g_i)$，其中$g_i$为满足 d 流的子图，即$\text{Flow}(g_i) \geq d$，$\text{Flow}(g_i)$为子图 g_i 能够满足的最大流

特别的，当 d 为最大流的时候，Pr 为随机流网络的最大流容量可靠性。本文在不特殊说明的情况下，容量可靠性都指的是最大流容量可靠性。

2.2.2 常见的随机流网络容量可靠性算法

目前已有的计算流网络容量可靠性的方法，大部分都采用基于 d-flows[25]和基于 d-cuts[29]的不同状态空间划分方法，将状态空间划分成能够满足流量阈值需求的合格状态集，不能够满足流量阈值需求的不合格状态集和同时包含了合格状态和不合格状态的待划分状态集，通过迭代的方式进一步处理待划分状态集，最终通过计算合格状态集中所有合格状态的概率之和来得到网络容量可靠性。

常用的状态搜索规则如下：

算法 2-3. 空间划分搜索算法 SDA

Input: G, souce s and sink t, Fmax
Output: Capacity reliability of max flow with s-t

1. init P = 0 , y← (0,0,⋯,$0_{|E|}$)， SET
2. push all the sub graphs states {(0,0,…,$0_{|E|}$),(1,1,…,$1_{|E|}$) , , I =0, j=0} into stack
3. **while** stack is not empty
4. get current closed space [α,β] of sub graphs according to j in the top state of stack
5. **if** j < I , then j++
6. **else** pop stack **End**
7. get the max flow value f_1 on MSG(space [α,β]) and f_2 on the lower(space [α,β])
8. **if** f_2 ≥ Fmax , **then**
9. SET← space [α,β]
10. **else if** f1<Fmax **then**
11. remove space [α,β];
12. **else if** f1≥Fmax and f_2 < Fmax **then**
13. use fv to get pivot sets IC and push {α, β, I_C ,| I_C |,1} into stack
14. get C_0 = [α',β] by {α, β, I_C ,| I_C |,1}

15. SET← C_0

16. **End**

17. **End**

18 **Return** SET

那么基于状态划分规则的随机流网络容量可靠性算法如下：

算法 2-4. 随机流网络容量可靠性算法

Input：G and it's souce s and sink t
Output: Capacity reliability Pr of max flow with s-t

1. Fmax = Dinic(MSG(G), s, t)

2. (subgraph SET) ← SDA(G, Fmax, s, t)

3. P_r← p_r of all subgraph that satisfy d flow

本文使用的容量可靠性一般指的是最大流对应的容量可靠性。使用的计算方法是使用状态划分算法获取所有满足最大流的子图，对应的容量可靠性就是所有满足最大流的子图概率之和。

2.3 不确定图数据

2.3.1 不确定图及可能世界模型

针对不确定图的研究，大多数是建立在标准的概率图模型基础之上的，即给定一个图 $G(V, E)$，其中 V 是顶点集，E 是边集，对任意的边 $e \in E$，e 都对应一个存在概率 p_e 和不存在概率 q_e，且满足 $p_e + q_e = 1$。如图 2-3 中的 G 是一个不确定图，边(V_2,T)上的数值 0.8 表示该边在 G 中存在的概率，此时边(V_2,T)容量为 2，对应的不存在概率为 0.2，即容量为 0。

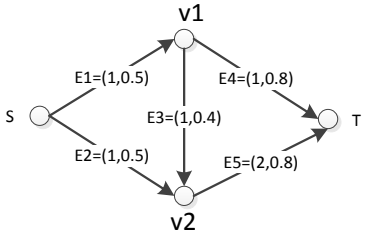


图 2-3 不确定图 G

当前，关于不确定图的理论和应用研究包括许多方面。Petteri Hinstsane[47]就如何寻找不确定图中最可靠子图的问题展开了深入研究，国内哈工大的邹兆年、张硕等人分别针对不确定图频繁子图模式挖掘和不确定图数据库的查询处理等问题进行了深入的分析与研究，就如何构建挖掘模型、选择 k 极大频繁子模式、如何降低子图同构计算的复杂度以及图数据库索引设计等问题，提出了许多高效可行的方法[42-45]。而对于不确定图的相邻性与可达性也有很多的研究成果，张海杰 [45]等人考虑不确定图中的 top-k 近邻查询、Potanmias[46]等人关于 k-NN 查询问题，都分别提出了相应的距离函数和高效可行的查询算法；然而，东北大学袁野[47]等人则关注于不确定图中给定顶点的可达性，提出了一种

有效的随机算法，此外，对于不确定图中的 top-*k* 生成树及相似性查询问题也有研究，许多有效的解决思路和算法[48] 被相继提出。

下面给出不确定图的相关概念：

定义 2-7.（不确定图）一个不确定图 G 是一个五元组 G=(V，E，s，t，(C，P))，其中，V 是有向图 G 中顶点的集合，E 是 G 中边的集合，s 和 t 分别为 G 的源点和汇点，(C，P)是一个二元组且 C：E->N 是边上容量函数，P：E->(0，1]是边上的概率函数，表明该边能通过的最大容量为 C 时的概率为 P，当边不存在，即边上能通过的容量为 0 时对应的概率为 1-P。

实例 2-1. 图 2-3 所示为一个不确定图 G，该不确定图的源点为 s，汇点为 t，除此之外还包含其他的顶点 v_1 和 v_2，边集为{E_1，E_2，E_3，E_4，E_5}，以边 E_4 为例，边上的容量 $c(E_4)=1$，概率 $p(E_4)=0.8$，也就是说 E_4 边能够达到流量为 1 的概率为 0.8，而容量是 0 的概率为 $1-p(E_4)=0.2$。

定义 2-8.（剩余不确定图）对于一个不确定图 G，如果有一条边 e'被移除，剩下的不确定图被称为原不确定图 G 移除边 e'之后的剩余不确定图 G'，则 G'可以表示为 G'=(V，E-e'，s，t，(C，P)-e' (c，p))，其中剩余不确定图的顶点和原不确定一致，且同时删除被移除边上的容量和概率的对应关系。

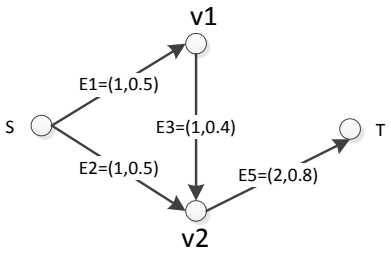


图 2-4 不确定图 G 在移除边 E_4 后的剩余不确定图 G'

图 2-4 为不确定图 G 移除边 E_4 之后的剩余不确定图 G'，G'中的顶点与原不确定图一致，同时 E_4 上的流量概率对应被移除。可能世界模型[49-50]是不确定图问题研究中的基本模型，它将不确定图扩展成为可能世界实例的集合，而每个可能世界实例都对应着一个概率值，且每个可能世界的实例可以被看成是一个确定图，这样就可以把不确定图中的相关问题转换为确定图问题。

对于一个给定的不确定图 G，其可能世界空间包含有 $2^{|E|}$个可能世界实例。对于图 2-3 中的不确定图 G 对应的可能世界空间大小为 $32 = 2^5$，每个可能世界实例 g 的概率 $P(g) = \Pi_{e \in E(g)} p_e \times \Pi_{e \in E \backslash E(g)} (1-p_e)$，其中：$E(g)$表示 g 的边集，$E \backslash E(g) = \{e \mid e \in E \wedge e \notin E(g)\}$。

2.3.2 不确定图分布可靠性

不确定图流分布问题是在传统的流网络问题的基础上加入了边的存在概率，因此对于不确定图流分布问题而言，边上存在着两种属性，一种是容量属性，另一种是概率属性。

定义 2-9.（不确定图的子图）不确定图 G=(V，E，s，t，(C，P))的子图 g(V'，E'，s，t，C')是一个确定图，其中 V'= V，E'∈E，C'是容量的集合，且 C'满足∀e∈E'，C' (e)=C(e)。如果 E' = E，则称 g 为不确定图 G 的最大子图，记作 MSG(G)。

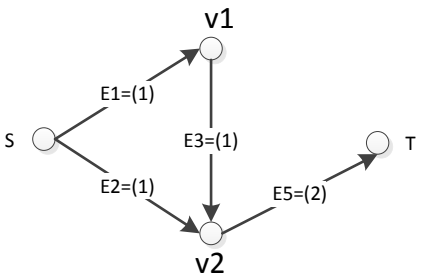


图 2-5 不确定图 G 的子图 g_1

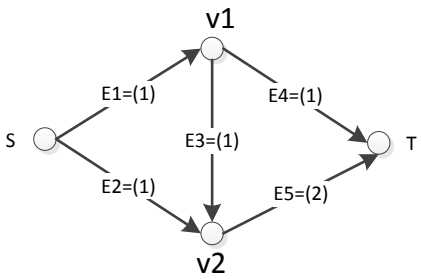


图 2-6 不确定图 G 的子图 g_2

如图 2-5 和图 2-6 所示，g_1 和 g_2 是图 2-3 中不确定图 G 的两个子图，根据定义子图 g_2 为不确定图 G 的最大子图 MSG(G)。

定义 2-10.（不确定图的子图概率）不确定图 G=(V，E，s，t，(C，P))的一个子图为 g(V'，E'，s，t，C')，则该子图 g 的概率为$P(G) = \prod_{e\in E'} P_e * \prod_{e\notin E'且\ e\in E}(1-P_e)$。即子图中边的存在概率和不确定图中其余边的不存在概率之积。

实例 2-2. 根据上述的子图概率计算的公式可得，不确定图 G 的子图 g1 的概率为 $P(g_1)$=0.5*0.4*0.5*0.8*(1-0.8)=0.016，同样的，可以知道其最大子图 g_2 的子图概率为 $P(g_2)$=0.5*0.5*0.4*0.8*0.8=0.064。

定义 2-11.（不确定图 s-t 流）给定源点 s 和汇点 t，不确定图 G 上的 s-t 流 f 是一个映射，它把每条边 对应到一个非负实数，f：E→R+；值 f(e)直观上表示由边 e 携带的流量。一个流 f 必须满足下面两个性质：

（1）（容量条件）对每条边 e∈E，有 0≤f(e)≤c(e)。

（2）（守恒条件）除了 s 和 t 之外，对每个结点 ，有$\sum_{e:\ e\to v} f(e) = \sum_{e:\ v\to e} f(e)$，即：所有进入结点 的流值之和等于所有从 出来的流值之和。

定义 2-12.（不确定图的最大流）不确定图 G 中，在每条边都存在时，能够从源点 s 到汇点 t 传输的最大流值 f_{max} 为不确定图 G 的最大流。

实例 2-3. 图 2-3 所示的不确定图 G 能够达到的最大流为 2。

定义 2-13.（不确定图的流分布及其可靠性）给定一个不确定图 G=(V，E，s，t，(C，P))及其上的一个流分布 F，则 F 的可靠性是不确定图 G 所有包含该流分布 F 的子图概率之和，可以表示为$P(F) = \sum_g \{p(g) | g \geq F\}$ 同时也等于不确定图 G 中所有流量不为 0 的边的概率之的边的概率之积，即$P(F) = \prod_{i=1}^{|E|} \{p(e_i) | f(e_i) > 0\}$。特别的，当流分布达到的流量为不确定图最大流时，为不确定图的最大流分布，对应的为不确定图的最大流分布可靠性。

实例 2-4. 如图 2-7 为不确定图 G 的 2 个最大流分布 F_1 和 F_2，分析可知，最大流分布 F_1 被子图 g_2 和 g_3 所蕴含，则 $P(F_1)=P(g_2)+p(g_3)$=0.16，此时也可以根据分布中流量不

为 0 的概率之积来计算，此时的计算过程为 $P(F_1)=0.5*0.5*0.8*0.8=0.16$，结果一致。同理，对于分布 F_2，通过定义可知 $P(F_2)=0.08$。

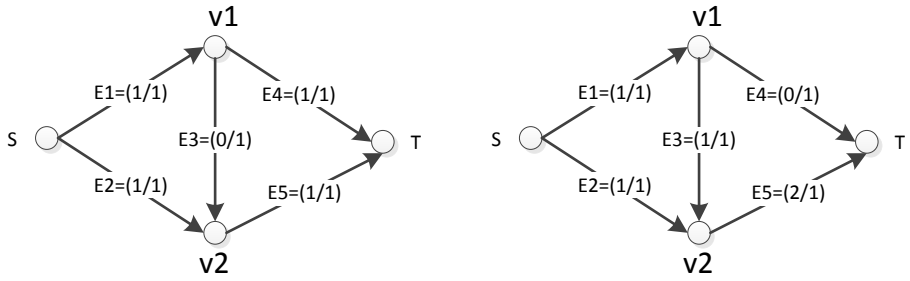


图 2-7 不确定图 G 的两个最大流分布 F_1 和 F_2

定义 2-14.（不确定图最可靠流分布）一个不确定图最可靠流分布是一个不确定图的流分布，其可靠性不小于该不确定图上的任一流量的流分布。特别的，当流量为最大流时，为最可靠最大流分布，其对应的可靠性为最可靠最大流分布可靠性。

实例 2-5. 如图 2-7 中的两个分布是不确定图 G 的两个最大流分布，其中 $P(F_1)>P(F_2)$，且不存在一个其他的最大流分布 F，使得 $P(F)>P(F_1)$，则 F_1 是不确定图 G 的最可靠最大流分布，其对应的可靠性为不确定图 G 的最可靠性最大流分布可靠性。

2.3.3 不确定图容量可靠性

定义 2-15.（不确定图的 d 容量可靠性）不确定图的 d 容量可靠性 P(d)可以表示为不确定图所有能够满足 d 流的子图概率之和，即$P(d)=\sum_g\{p(g)|f(g)\geq d\}$，特别的，当 d 为不确定图的最大流时，表示的为不确定图相对于最大流的容量可靠性。

实例 2-6. 如图 2-8 为图 2-3 所示不确定图 G 的三个能够满足最大流 2 的所有子图，那么不确定图的最大流的容量可靠性可以表示为 $P=p(g_1)+p(g_2)+p(g_3)=0.176$。

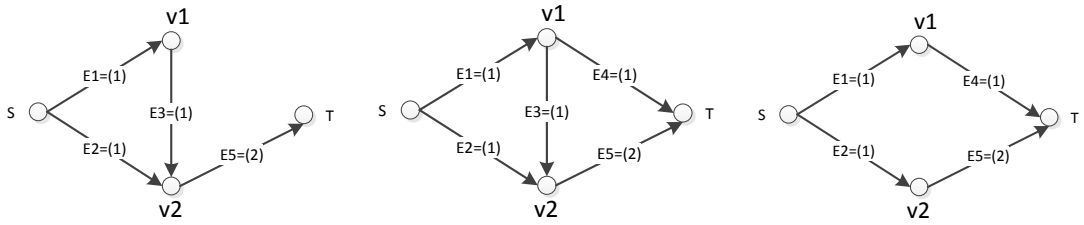


图 2-8 不确定图 G 的三个能够满足最大流 2 的所有子图 g_1，g_2，g_3

2.4 关键节点计算方法分析

（a）基于元件局部环境的角度

本类方法是最简单直观的方法，度中心性考察节点的直接邻居数目，半局部中心性考虑了节点 4 层邻居的信息。k-壳分解可以看作度中心性的一种扩展，它根据节点在网络中的位置来定义其重要性，认为越是在核心的节点越重要。

（b）基于元件所处位置的角度

在交通、通信、社交等网络中存在着一些度很小但是很重要的节点，这些节点是连接几个区域的“桥节点”，它们在交通流和信息包的传递中担任重要的角色。此时，刻

画节点重要性就需要考察网络中节点对信息流的控制力，这种控制力往往与网络中的路径密切相关。另一方面，从提高网络的可靠性和抗毁性角度看，任意节点对之间的路径数目越多，网络的鲁棒性就越高。

（c）基于元件对网络功能的影响角度

网络的结构会处于动态变化之中，节点的重要性往往体现在该节点被移除之后对网络的破坏性。从衡量网络的健壮性角度看，一些节点一旦失效或移除，网络就有可能陷入瘫痪或者分化为若干个不连通的子网。实际生活中的很多基础设施网络，如输电网、交通运输网、自来水-天然气供应网络等，都存在“一点故障, 全网瘫痪”的风险。但由于计算复杂度较高，目前这类方法还仅限于小规模的网络实验。

2.5 本章小结

本章主要介绍了一些基本概念。由于本文是基于网络最大流、随机流网络可靠性和不确定图分布可靠性对不确定图的边关键度进行的相关研究，因此主要介绍这些因素的相关研究。首先介绍了最大流的相关概念和常用的算法，然后介绍了随机流网络可靠性的基本概念，最后介绍了不确定图的相关概念和不确定图数据的一般建模方法，以及不确定图流分布及其可靠性的相关问题。本节的最后介绍了目前常用网络关键节点的算法。

第 3 章 基于流量和容量可靠性的关键边衡量方法

本章从边移除（故障）后对最大流和（最大流）容量可靠性产生的相对损失这一角度，构建了一种基于流量和容量可靠性的关键边衡量模型，对不确定图边的关键度进行综合评估。针对此模型，本章首先设计了一种基于最大流和容量可靠性重新计算的原始算法 BASE_FCP，然后对于其性能存在的不足，提出一种基于流量和容量可靠性的状态区间缓存算法 SCA_FCP。该算法首先根据移除边之后，对于动态不确定网络最大流和容量可靠性的影响，将边定性分类，然后根据不同的边设计不同的计算方式获取容量可靠性，以减少复杂度。本章最后，比较 BASE_FCP 算法和 SCA_FCP 算法在空间复杂度和时间复杂度上的差别。

3.1 基于流量和容量可靠性的关键边模型

对于一个网络而言，能够达到的最大流的大小，体现出了网络的联通能力和传输能力，例如，网络的最大流受限于网络的最小割集，也就是说最小割集中的任意一条边断掉都会使得整个网络可以传输的最大流下降，这样的边可以看做是整个网络相对于流量的关键边。在不确定图中，相对于可靠性的关键边对网络信息传输的可靠性起到关键作用，这样的环节一但发生故障，会使得整个网络的可靠性下降。

本章构建了基于流量和容量可靠性作为指标的不确定图的关键边评估模型，针对不确定图中边移除（故障）后对流量和容量可靠性产生的相对损失，来评估关键边。该模型将流量作为衡量关键边的最重要因素，当流量一致时，比较（最大流）容量可靠性。假设 f，p_c 分别表示不确定图边移除之后剩余不确定图能够达到的最大流和最大流容量可靠性，K_f，K_{pc} 分别表示为流量和容量可靠性的关键系数，那么模型中的指标关键程度可以表示为 $K_f>K_{pc}$。

根据这两个指标，可以将不确定图的边分为三类，即边发生故障之后剩余不确定图和原不确定图进行对比：

（1）流量和容量可靠性都减少，作为 A 类边；

（2）流量不变，容量可靠性减少，作为 B 类边；

（3）流量和容量可靠性都不变，作为 C 类边。

根据这两个指标的关键程度 $K_f>K_{pc}$，易得出 ABC 三类边的关键程度依次减少。

首先，通过状态划分规则[27]可以获取不确定图满足最大流的所有子图区间$\{C_1,C_2,...,C_\tau\}$；然后，根据$\{C_1,C_2,...,C_\tau\}$中包含 e_i（且满足最大流）的子图个数，对边 e_i 进行分类。根据不确定图子图的向量表示法[27]以及向量的偏序关系，本文在获取的子图区间的基础上定义包含 e_i（且满足最大流）的子图个数 S_i。

为了获取 S_i，首先定义 s_{ij} 为边 e_i 在子图区间 C_j 的存在状况，其中 $0\leqslant j\leqslant\tau$（τ为满足最大流的区间个数）。根据偏序规则，其中 $e_i=1$ 表示在子图区间 C_j 中的所有子图都包含

边 e_i；e_i=0 表示在子图区间 C_j 中的所有子图都不包含边 e_i；如果 e_i=x 则表示子图区间 j 中的子图有包含和不包含两种情况，定义 s_{ij} 如公式 3-1 所示：

$$s_{ij} = \begin{cases} 1 & (e_i = 1) \\ 0 & (e_i = x,0) \end{cases} \tag{3-1}$$

其中 i 表示边的序号，j 表示子图区间的序号。对于不确定图中的每一条边 e_i 都有边存在个数 S_i，定义如公式 3-2 所示：

$$S_i = \sum_{j=1}^{\tau} s_{ij} \tag{3-2}$$

其中 i 为边的序号，i∈{1….n}，n 为不确定图 G 边的个数，j 为子图区间的序号，j∈{1…τ}，τ 为使用状态划分后闭合区间的个数。

根据定义可知，S_i 有性质$0 \le S_i \le \tau$，其中τ为使用状态划分区间算法获得的最大流区间的个数。

根据边存在率 S_i 的定义可知，如果 S_i=τ，那边边 e_i 存在于所有的满足最大流子图中，因此当 e_i 发生故障时，所有满足最大流的子图都遭到破坏而不能满足最大流，从而相应的最大流容量可靠性也发生改变，因此当对于边 e_i，如果 S_i=τ，那么 e_i 为 A 类边；同理如果 0＜S_i＜τ，表示边 e_i 存在于部分，满足最大流的子图中，那么当边 e_i 故障之后，依然有剩余的一些满足最大流的子图可以不被破坏，这时依然可以达到最大流，但是最大流容量可靠性降低，此时边 e_i 为 B 类边。

实例 3-1. 对于图 2-3 中的不确定图 G，其经过状态划分算法获取所有满足最大流的区间表示如表 3-1 所示，根据公式 3-1，可知对于边 e_1，有 s_{11}=1，s_{12}=1，根据公式 3-2 可知，对于边 e_1 的边存在率 S_1=s_{11}+s_{12}=2=τ，所以对于不确定图 G 而言，边 e_1 是一个 A 类边，同理可以知道边 e_1，e_2 和 e_5 为 A 类边，e_3 和 e_4 为 B 类边。

表 3-1 不确定图 G 所有满足最大流的区间

编号	满足最大流的区间
1	11x11
2	11101

综上所述，根据边存在率 S_i，对边进行定性的分类如下：

1、S_i=τ ，那么 e_i 为 A 类边；

2、0＜S_i＜τ，那么 e_i 为 B 类边；

3、S_i=0，那么 e_i 为 C 类边；

根据以上的定性分析，可以得到如下的一系列的推论：

推论 3-1. 对于不确定图 G 中的一条边 e_i，如果有边 e_i 对应的 S_i=τ，那么移除边 e_i 会使得不确定图不能满足原有最大流和容量可靠性。

证明 3-1. 想要证明移除 A 类边，会使得不确定图 G 最大流减少，需要证明两点（1）A 类边移除，不确定图 G 不能达到最大流；（2）非 A 类边移除，不确定图仍然可以达到最大流。下面将按照这两条分别证明：

（1）因为 e_i 是 A 类边，有 $S_i=\tau$，根据定义，边 e_i 存在于不确定图所有的满足最大流的子图中，假设经过划分保存下来的所有闭合子图空间集为$\{C_1, C_2, \ldots, C_\tau\}$，满足 $C_i \cap C_j = \varnothing, C_1 \cup C_2 \cup \ldots \cup C_\tau = C$，其中τ 为满足最大流的子图区间的个数，因为 $S_i=\tau$，所以对于满足最大流的子图 g，一定有 $e_i \in g$ 且 $g \in C$，如果当 e_i 发生故障之后，剩余不确定图G'的任意子图g'都不包含 e_i，即$e_i \notin g'$，因此对于 e_i 发生故障后的剩余不确定图G'的任意子图都不能达到最大流，那么G'也不能达到最大流；

（2）同理，对于一条非 A 类边，因为 $S_i<\tau$，则在 C 中必存在一个子图 g，使得$e_i \notin g$，如果边 e_i 发生故障，不确定图可以使用子图 g 传递流量，同时因为 $g \in A$，即 g 能传递最大流，所以对于非 A 类边发生故障，不确定图 G 仍然能够达到最大流。

实例 3-2. 对于图 2-3 中的不确定图 G，经过状态划分过程，获取的所有满足最大流区间如表 3-1 所示，因为经过划分有两个区间，则τ=2，根据公式 3-1，可知对于边 e_1，有 $s_{11}=1$，$s_{12}=1$，根据公式 3-2 可知，对于边 e_1 的边存在率 $S_1=s_{11}+s_{12}=2=\tau$，所以对于不确定图 G 而言，边 e_1 是一个 A 类边，A 发生故障使得不确定图的最大流从原来的 2 下降为 1。

实例 3-3. 对于图 2-3 所示的不确定图 G，经过状态划分过程，获取的满足最大流的区间分别是 11x11 和 11101，其中τ=2，根据公式 3-1，可知对于边 e_3，有 $s_{31}=0$，$s_{32}=1$，根据公式 3-2 可知，对于边 e_3 的边存在率 $S_1=s_{11}+s_{12}=1<\tau$，所以对于不确定图而言 e_3 是一个 B 类边，则 e_3 移除，使得子图 11111 和 11101 不能满足最大流，但是 11011 依然满足最大流，此时的（最大流）容量可靠性下降，仅为 11011 子图的可靠性。

推论 3-2. 对于不确定图中的一条边 e_i，如果有边 e_i 对应的 $S_i=0$，那么移除边 e_i，不确定图的最大流，（最大流）容量可靠性都不发生改变。

证明 3-3. 若要证明对于不确定图 G 中的任意一条边 e_i，如果 e_i 是 C 类边，即如果 $S_i=0$，则 e_i 断掉之后，不确定图的最大流和容量可靠性都不发生改变，只需要证明（1）C 边断掉之后，不确定图 G 的最大流不变（2）C 边断掉之后，不确定图 G 的容量可靠性不变。下面将对这两点分别证明：

（1）对于不确定图 G，经过划分算保存下来的所有闭合子图空间集为$\{C_1, C_2, \ldots, C_\tau\}$，满足 $C_i \cap C_j = \varnothing$，对于不确定图 G 中的 C 边 e_i 有 $S_i=0$，即当 e_i 发生故障之后，至少存在一个子图 $g \in C_1 \cup C_2 \cup \ldots \cup C_\tau$且$e_i \notin g$，所以当 e_i 移除之后，依然有 g 满足最大流，即 e_i 边断掉之后，不确定图 G 的最大流不变；

（2）首先说明当 $S_i=0$，那么对于所有的子图区间每一个对应的边 ei 处都为标记“x”，即任意一个子图区间可以表示为 $C=(c_1, c_2 \ldots x \ldots c_n)$，其中 c_i 表示为子图区间 C 的各条边对应的表示，$i=\{1 \ldots i-1, i, i+1 \ldots n\}$，则 C 可以分为两个子图区间（或子图）$C_1=(c_1, c_2 \ldots 0 \ldots c_n)$和 $C_2=(c_1, c_2 \ldots 1 \ldots c_n)$，那么在移除边之前，区间 C 的可靠性可以表示为 $P=p(c_1)*\ldots.*p(e_{i-1})*p(e_i=0)*p(e_{i+1}) \ldots *p(c_n)+p(c_1)*\ldots.*p(e_{i-1})*p(e_i=1)*p(e_{i+1}) \ldots *p(c_n)= p(c_1)*\ldots.*p(e_{i-1})*p(e_{i+1}) \ldots *p(c_n)*(p(e_i=0)+ p(e_i=1))$，又因为 $p(e_i=0)+ p(e_i=1)=1$，所以 C 边断掉之后，不确定图 G 的容量可靠性不变。

实例 3-4：假设对于一个不确定图，其经过划分获取的子图区间只有一个，11x11，那么根据公式 3-1 和公式 3-2，边 e_3 为 C 类边，因此 e_3 移除之后，依然可以满足最大流。移除 e_3 之前，容量可靠性 $p_c=p(e_1)p(e_2)(1-p(e_3))p(e_4)p(e_5)+ p(e_1)p(e_2)(e_3)p(e_4)p(e_5)= p(e_1)p(e_2)p(e_4)p(e_5)$，移除 e3 之后，容量可靠性表示为 $p_c'= p(e_1)p(e_2)p(e_4)p(e_5)$，显然移除 e_3 之后不确定图的容量可靠性不变。

3.2 基于重新计算的BASE_FCP算法

基于重新计算的基础算法 BASE_FCP（baseline algorithm with flow and capacity relia -bility）的思想是，首先计算不确定图每一条边移除之后能够满足的最大流，然后通过比较不确定图在移除边之后能够达到的最大流，只有当流量相同的情况下，才计算比较容量可靠性，本章将 BASE_FCP 算法作为基于流量和容量可靠性模型的关键边基本算法。

根据上述分析，下面给出重复计算的基础代码实现（伪代码）：

算法 3-1. 基于重新计算的 BASE_FCP 算法

Input: uncertain graph G and it's source and sink

Output: sorted key edge set

1. Input uncertain Graph G and its source node and sink node
2. ALLEdgeFlowDecrease[|E|] ← Get the max flow of the remain uncertain graph when every single edge being cut off by Dinic
3. Sort ALLEdgeFlowDecrease[|E|] by max flow
4. i←0, j←1
5. **While** j < |E| **then**
6. **if** the max flow of the remain uncertain graph ($G-e_i$) equles the max flow of ($G-e_j$) **then**
7. recompute the p_c of Uncertain Graph $G(e_{i=0})$ by SDBA[27]
8. **End**
9. i←j, j←j+1
10. **End**
11. Sort ALLEdgeFlowDecrease[|E|] by p_c
12. **Return** sorted key edge set

基于重新计算的 BASE_FCP 算法首先要获取所有边发生故障之后能够满足的最大流，然后对计算的流量进行排序，当前后两条边断掉之后获得的流量不相等时，根据模型定义，直接就可以比较出两条边的关键程度（流量相对少的更关键），而只有当两条边断掉获得流量相等的情况下，才会重复计算不确定图的容量可靠性，此时使用容量可靠性来区分这两条边的关键程度。

基于重复计算的基础算法运行效率与重复计算的次数紧密相连，算法首先需要计算所有边在发生故障之后对应的最大流，本文使用最大流算法是 Dinic 算法，运行 Dinic 算法所需的处理时间为 $O(|V|^2|E|)$，因此在处理这部分是复杂度是 $O(|V|^2|E|^2)$；然后当移除

边之后，在剩余图最大流相同的情况下需要计算容量可靠性，本文使用的重复计算方式是基于状态划分的 d-下界点计算方式，划分过程复杂度为 $O(k|V|^2|E|)$，其中 K 为划分过程中需要运行 Dinic 算法的次数，在状态划分结束之后，根据容量可靠性的定义，需要获取所有满足最大流的子图概率之和，这一部分的复杂度可以表示为 $O(T|E|)$，其中 T 为满足最大流子图的个数。因此整个 BASE_FCP 算法的复杂度为 $O(km|V|^2|E|) + O(|V|^2|E|^2)+ O(T|E|)$，其中 m 为需要重复计算的次数，一般 m 的范围是 $0 \leqslant m \leqslant |E|$。

3.3 基于流量和容量可靠性的关键边算法SCA_FCP

3.3.1 算法思想

虽然 BASE_FCP 算法对于移除边之后最大流不一致的情况，减少了一定量的容量可靠性的重复计算，但在图规模较大的情况下，重复计算的次数可能变的很多，这显然对于大规模图或者高稠密图是不适用的。本节提出了一种基于流量和容量可靠性的状态区间缓存算法 SCA_FCP（state caching algorithm with flow and capacity reliability），该算法的主要思想是根据模型中不同类型的边采用不同的算法，以减少计算复杂度。当 A 边移除，使用关于 A 类边的子算法 STPA_FCP（state tree pruning algorithm with flow and capacity reliability），在区间状态树上，利用割集中的边剪枝获取移除该边之后，剩余不确定图满足最大流的所有子图集合，其子图概率之和为（最大流）容量可靠性；当 B 边移除，使用关于 B 类边的子算法 BSA_FCP（B class edge search algorithm with flow and capacity reliability）获取剩余不确定图依然满足最大流的区间集合；因为 C 边移除之后不影响，所以不需要计算。

基于以上思想，SCA_FCP 算法可以表示为如图 3-1 步骤：

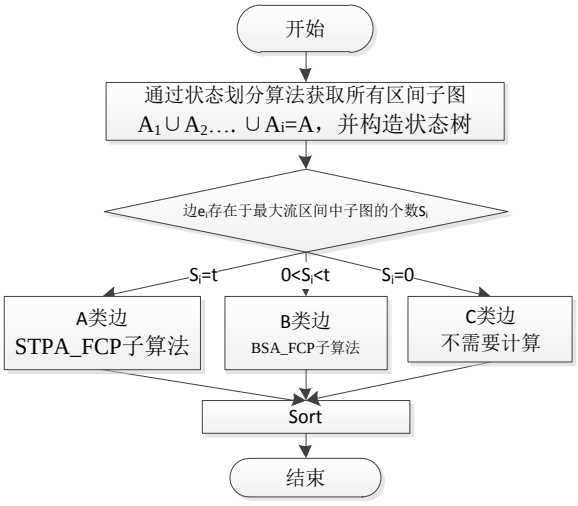


图 3-1 算法 SCA_FCP 流程图

3.3.2 算法步骤

（I） A 类边的子算法 STPA_FCP

本文采用状态划分算法获取不确定图所有满足最大流的子图区间。初始图在状态划分的过程中，可以生成一个区间状态树，这样的一个区间状态树生成如算法 3-2 所示：

算法 3-2. 基于状态划分算法生成区间状态树过程

Input: uncertain graph G and it's source and sink

Output: created tree T

1. init stack S ← C[L, U], L=(0000…000), U=(111…111)
2. set C[L, U] as the root node of T
3. **While** S is not empty then
4. C ← get the top of S and pop S
5. check the lower bound and upper bound of C with DINIC
6. **If** lower(C) < max_flow and upper(C) = max_flow **then**
7. get sub-state of C with SDBA[27] as (C_1,C_2…C_n) as the child node of C and push (C_1,C_2…C_n) into S
8. **Else if** lower(C) > max_flow **then**
9. Set C as **ACCEPT** state;(all the subgraph in C satisfy max flow)
10. **Else if** upper(C) < max_flow **then**
11. Set C as **DENY** state;(all the subgraph in C not satisfy max flow)
12. **End**
13. **End**
14. **return** T

如图 3-2 所示，为图 2-3 中不确定图 G 的状态划分过程，其中斜体下划线的状态为未知状态，需要继续划分，加粗字体的子图区间为满足最大流的状态区间，而其他的区间为不满足最大流区间。

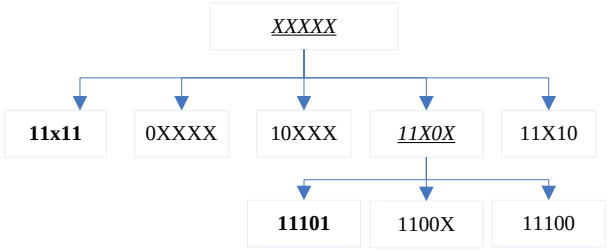


图 3-2 完全状态划分树 State-Tree

对于 A 类边，由推论 3-1 可知，其断掉会使流量减少，即不能达到原有的最大流，此时原有最大流的容量可靠性下降为 0，需要计算剩余不确定图的新最大流（小于原有最大流）的容量可靠性。

在 State-Tree 中将原有的该边位置为 1 状态的去除，那么树中留下来的区间是移除该边之后的所有子图区间。

实例 3-5. 图 3-2 为不确定图 G 的完全状态划分树，对于不确定图 G 的 A 边 e_5，移除之后按照以上的处理方式，获取的状态树如下图 3-3 所示，所有的叶子节点为移除 e_5 之后的子图空间的划分。

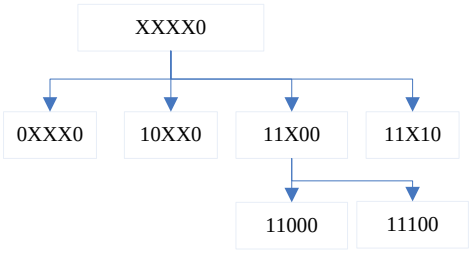


图 3-3 移除 e5 之后的状态区间树

对于某一条 A 边而言，树中的每一个区间该边位置都采用 0，目的是获取不相交的区间。对于新构成的一个子图树，本节提出一种基于割集的状态树剪枝算法 STPA_FCP，该算法主要是利用了割集中的边必定在满足最大流子图这一性质，通过割集中的边对于子图状态树进行剪枝，以达到减少搜索的目的。

该算法首先获取根节点，使用最大子图 MSG 通过 DINIC 获取割集边，对于其所有子节点，通过割集中的边进行剪枝，因为满足最大流的子图必定包含割集中的边，所以不包含割集中边的中间节点和叶子节点都必须舍去，对于保留的子图区间，进行进一步划分即可，即

（1）当区间的下界子图满足最大流，则整个区间满足最大流，保留区间；

（2）当区间上界不满足最大流，则整个区间不满足最大流，舍弃区间；

（3）区间下界不满足最大流，上界满足最大流，则需要二次划分区间。

通过这样的方式便可以有效的获取移除边 e 之后，剩余不确定图所有满足最大流的区间子图，进而计算移除 e 之后不确定图的容量可靠性。

实例 3-6. 如图 3-4 是经过割集剪枝之后的子图树。以为 e_1 和 e_4 为割集边，那么有背景的圆部分及其子树都被剪枝掉。

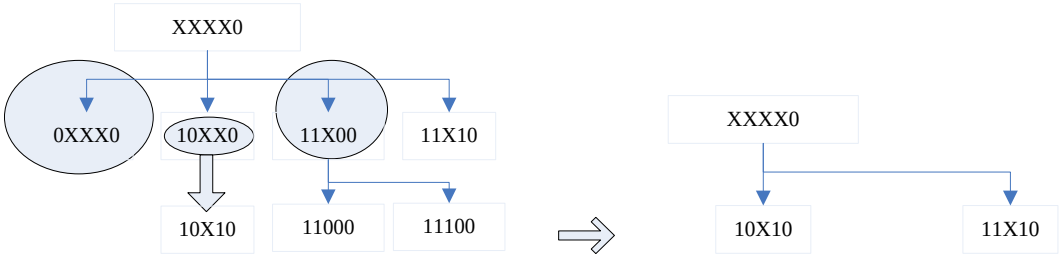


图 3-4 经过剪枝后的子图树

综上所述，STPA_FCP 算法的伪代码可以表示如下：

算法 3-3. 基于割集的状态树剪枝算法 STPA_FCP

Input: uncertain graph G and it's source and sink, the breaked A class edge e, subgraph section tree T

Output: The capacity reliability p_c of uncertain graph (G-e)

1. get cut edge set by DINIC of MSG(G-e) and init SET
2. init stack S and push all the left ode of T (which pruned by cut edge set) into S
3. **While** S is not empty， **then** pop S as C
4. **If** lower(C) = F_{max} **then** push C into SET

5. **If** upper(C) < F_{max} **then** move out C
6. **If** lower(C) < F_{max} and upper(C) = F_{max} **then**
7. get sub-state of C with SDBA[27] as (C_1,C_2…C_n) and push (C_1,C_2…C_n) into Stack S
8. **End**
9. **End**
10. p_c ← summary probability of subgraph in SET
10. **Return** p_c

（II） B 类边搜索算法 BSA_FCP

当 B 类边断掉之后，剩余不确定图的最大流不会减少，此时的重点在于计算剩余不确定图的容量可靠性，即移除 B 边之后，筛选出原有满足最大流的子图集合中，哪些子图依然满足最大流，且不能存在重复的子图，然后通知子图概率之和获取对应最大流容量可靠性。

本节提出算法 BSA_FCP，用于计算获取 B 边移除之后，依然能够满足最大流的所有子图。该算法步骤如下：

算法 3-4. B 类边搜索算法 BSA_FCP

Input: The subgraph (that satisfy maximum flow) section of uncertain graph G and B class edge e_i

Output: The capacity reliability p_c of uncertain graph (G-e_i)

1. init set, stack S
2. push the subgraph (that satisfy maximum flow) section of uncertain graph G into S
3. **While** S is not empty **then**
4. C ← get the top S and pop S
5. **If** the position of e_i in C is 1，**then** move out C
6. **If** the position of e_i position in C is 0，**then** push C into set
7. **If** the position of e_i in C is X，**then**
8. divide C to $C_{ei=1}$={e_1,e_2,…,e_{i-1},1,e_{i+1},…e_n} and $C_{ei=0}$= {e_1,e_2,…,e_{i-1},0, e_{i+1},…e_n}, then push $C_{ei=0}$ and $C_{ei=1}$ into S
9. **End**
10. **End**
11. p_c ← summary probability of subgraph in set
12. **Return** p_c

需要证明以上算法获取的是不确定图的子图是移除 B 边之后剩余不确定图所有满足最大流的子图，且不存在重复的子图。证明过程如下：

证明 3-4. 对于不确定图 G，经过划分后获取的满足最大流的所有区间为{C_1，C_2，…，C_τ}，满足 $C_i \cap C_j = \varnothing$，$C_1 \cup C_2 \cup \ldots \cup C_\tau = C$，其中 τ 为满足最大流的子图区间个数，

假设边 e_i 为 G 的 B 类边，根据 B 类边的定义，存在一个 k∈{1，2…τ}，使得 $\{C_1 \cup C_2 \ldots \cup C_k\} = C_{e_i=0}$ 与 $\{C_{k+1} \cup C_{k+2} \ldots \cup C_\tau\} = C_{e_i=1}$，令$C' = (C_1 - e_i) \cup (C_2 - e_i) \ldots \cup (C_k - e_i)$，因为$C_{e_i=0}$中的所有子图$e_i$位置为 0，所以有$C' = C_{e_i=0}$

若要证明C'为 G-e_i之后，所有满足最大流的子图集合，需要证明以下两点：

（1）$\forall g \in C'$，都有$f(g) = f_{max}$；

（2）$\forall g \in C_{e_i=1}$，若$f(g - e_i) = f_{max}$，则必存在一个$g_0 \in C'$，使得$g_0 = g - e_i$。

首先证明（1）：已知条件$\forall g \in C_{e_i=0}$都有$f(g) = f_{max}$，且$C' = C_{e_i=0}$，所以可以推出对于任意的子图 g，如果$g \in C'$，都有 g 满足最大流，即$\forall g \in C'$，都有$f(g) = f_{max}$。

然后证明(2)：反正法。假设$\forall g \in C_{e_i=1}$，若有$f(g - e_i) = f_{max}$，则对于$\forall g_0 \in C'$，都有 $g - e_i \neq g_0$。

对于$g \in C_{e_i=1}$，$g - e_i = \{e_1, e_2, \ldots, e_{i-1}, 0, e_{i+1}, \ldots, e_{|E|}\}$，且有$f(g - e_i) = f_{max}$，所以$g - e_i \in C_{e_i=0}$，又因为$C' = C_{e_i=0}$，所以$g - e_i \in C'$，这与不存在一个$g_0 \in C'$使得 $g - e_i = g_0$矛盾。假设不成立。

实例 3-7. 对于如图 2-3 中的不确定图 G，经过状态划分获取满足最大流的子图区间有两个，分别是 11x11 和 11101，e_3 是一个 B 边，根据以上规则，11x11 区间在 e_3 位置上的状态是 x，可以分为 11111 和 11011 两个区间，因为 11111 和 11101 两个区间在 e_3 位置上的状态是 1，所以舍去，而只保留区间 11011。所以最终移除 e_3 之后，依然满足最大流的子图区间是 11011。

对于 C 类边的计算，根据推论 3-2，其断掉不会影响不确定图的最大流和容量可靠性，也就是说 C 类边断掉不会对不确定图的状态造成任何的影响，因此不需要计算。

基于以上处理方式，包含以上子算法的 SCA_FCP 算法的伪代码如下：

算法 3-5. 基于流量和容量可靠性的关键边算法 SCA_FCP

Input: uncertain graph G and it's source and sink

Output: sorted key edge set

1. Input uncertain Graph G and its source node and sink node
2. maxFlow, StateMtrix, StateTree ← algorithm SDBA[27]
3. ALLEdgeFlowDecrease[|E|] ← Get the max flow of the remain uncertain graph when every single edge being cut off by Dinic
4. Sort ALLEdgeFlowDecrease[|E|] by max flow
5. Classify ABC edge by StateMtrix
6. i←0, j←1
7. **While** j < |E| **then**

8. **if** the max flow of the remain uncertain graph ($G-e_i$) equles the max flow of ($G-e_j$) **then**
9. check the class of e_i by step 5
10. **If** e_i is A class edge **then** use algorithm STPA_FCP to get p_c of ($G-e_i$)
11. **If** e_i is B class edge **then** use algorithm BSA_FCP to get p_c of ($G-e_i$)
12. **If** e_i is C class edge **then** p_c of ($G-e_i$) is same as G
13. **End**
14. i←j, j←j+1
15. **End**
16. Sort ALLEdgeFlowDecrease[|E|] by p_c
17. **Return** sorted key edge set

3.3.3 算法复杂度分析

SCA_FCP 首先需要获取不确定图所有边分别移除之后能够满足的最大流，然后对获取的流量进行排序，当前后两条边断掉之后获得的流量不相等时，根据模型定义，直接就可以比较出两条边的关键程度。当两条边移除之后剩余不确定图最大流一致，首先判断移除边的类型，然后根据三类边的不同性质，对于不同性质的边选择不同的算法获取移除该边之后剩余不确定图的容量可靠性，从而简化计算过程。

首先对于 A 类边，考虑最坏情况，复杂度主要体现在 K 次运行 Dinic 算法，因此对于 A 边的复杂度为 $O(ak|V|^2|E|)$，其中 a 为需要计算的 a 边个数，A 边计算结束，根据容量可靠性的定义，需要获取所有满足最大流的子图概率之和，这一部分的复杂度可以表示为 $O(aT|E|)$，其中 T 为满足最大流子图的个数；对于 B 类边，需要遍历满足最大流的所有区间，其复杂度为 $O(b\tau)$，其中 b 为不确定图 B 边的个数，τ 为满足最大流区间的个数，对于 C 边来说，因为不需要计算，只需要保留原有的最大流和容量可靠性，所以复杂度为 $O(c)$，其中 c 为不确定图 C 类边的个数。综上所述，基于状态划分树的算法的整体复杂度是 $O(|V|^2|E|^2)+O(ak|V|^2|E|)+ O(aT|E|)+ O(b\tau)+O(c)$，其中 $a+b+c=|E|$。

3.4 实验及分析

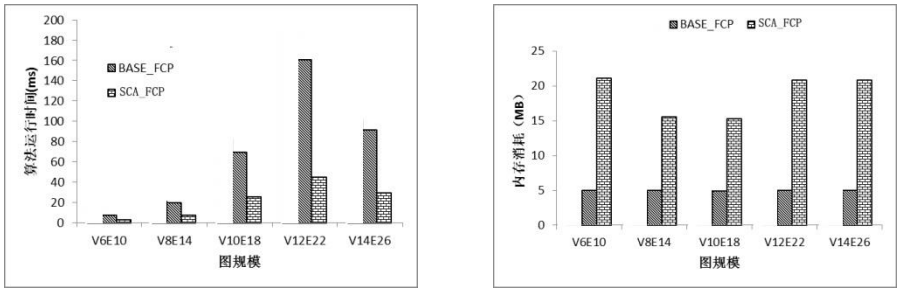
为了验证本文所提出算法的运行效率及分析影响算法性能的各种因素，本节进行了一系列的实验，实验平台为一台 Intel Core 的 PC 机（CPU i7-3770，3.40GHz，内存 8GB，64 位 windows 7 操作系统），所有算法采用 C++在 VS2010 上实现。

3.4.1 实验数据

本文采用了与文献[51]相同的数据集，使用 NETGEN 生成器产生 V_6E_{10}、V_8E_{14}、$V_{10}E_{18}$、$V_{12}E_{22}$、$V_{14}E_{26}$ 共 5 组不同图规模的二态有向图集合（本文实验数据集合大小为 5），其中 V_nE_m 表示有 n 个顶点、m 条边组成的图，图中边的容量与对应概率满足均匀分布。通过在不同图规模的情况下，比较算法 BASE_FCP 和 SCA_FCP 算法在运行时间及内存消耗放方面的差异。

3.4.2 实验结果与分析

实验 3-1. 不同图规模对于算法性能的影响



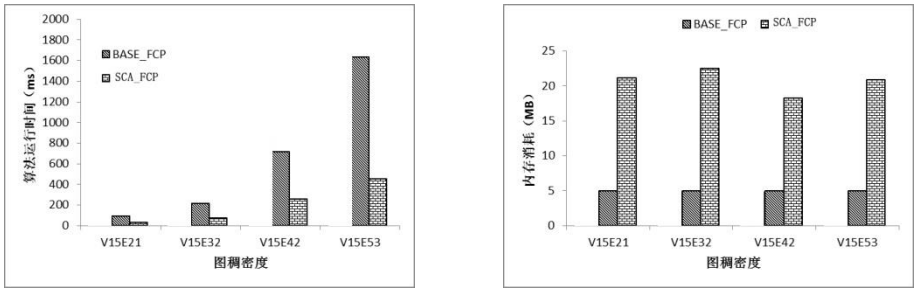
（a）时间消耗　　（b）内存消耗

图 3-5 不同图规模情况下 BASE_FCP 和 SCA_FCP 算法时间和内存消耗

根据图 3-5 所示，BASE_FCP 算法是在当流量一致的情况下才计算容量可靠性。如图 3-5（a）所示，SCA_FCP 算法在运行时间上有了很大的减少，适应性更好。如图 3-5（b)所示，SCA_FCP 算法相对于 BASE_FCP 算法在空间复杂度上有了一定程度的增加，但是随着图规模的增加，内存的使用并没有增加太多，依然在可以接受的范围内。

实验 3-2. 不同图稠密度对于算法性能的影响

为更好的反映算法 BASE_FCP 算法和 SCA_FCP 算法的性能差异，使用 NETGEN 生成器生成 $V_{15}E_{21}$，$V_{15}E_{32}$，$V_{15}E_{42}$，$V_{15}E_{53}$ 四种不同稠密度的图，通过在不同图稠密度的情况下，比较算法 BASE_FCP 和 SCA_FCP 算法在运行时间及内存消耗放方面的差异。试验结果如下：



（a）时间消耗　　（b）内存消耗

图 3-6 图稠密度对 BASE_FCP 和 SCA_FCP 时间和内存消耗影响

如图 3-6 所示，相对于不同稠密度的图，SCA_FCP 算法相对于 BASE_FCP 算法在空间复杂度上有一定的增加，但是时间复杂度上有较大的优势。

在实验的过程中，发现对于不确定图使用状态划分来求解所有满足最大流的子图区间的过程中，对于一块连续的满足最大流的区间，如果“划分线”（满足最大流的子图）获取的不是该区间的下界，那么最后获取的子图区间是有可能合并的。这样使得原本是 C 类边被归结为 B 类边，但是这不影响算法的正确性，因为 C 类边不需要计算，这样的结果是的算法的复杂度变高一点，但是由于 BSA_FCP 算法复杂度很低，所有依然是可以接受的。

3.5 本章小结

针对不确定图关键边的衡量，本章节构建了一种基于流量和容量可靠性的关键边衡量模型，使用不确定图中边移除（故障）后对流量和（最大流）容量可靠性产生的相对损失这一角度，对边的关键度进行综合评估。对于构建的基于流量和容量可靠性的模型，本章节首先设计了一种基于重复计算的基本算法 BASE_FCP。因为 BASE_FCP 算法复杂度较高，本章提出一种基于流量和容量可靠性的状态区间缓存算法 SCA_FCP，并重点介绍了算法的思想与实现，同时对算法进行了时间复杂度和空间复杂度的分析。最后通过实验分析，比较 BASE_FCP 算法和 SCA_FCP 算法在空间复杂度和时间复杂度上的差别，实验结果表明 SCA_FCP 算法相对于 BASE_FCP 算法其空间复杂度有一定的增加，但是时间复杂度方面具有较大的优势。

然而根据容量可靠性的定义，计算最大流容量可靠性需要获取不确定图所有满足最大流的子图，计算容量可靠性本身是一个非常复杂的过程。

第 4 章 基于流量和分布可靠性的关键边衡量方法

虽然基于流量和容量可靠性的模型对于关键边的区分度较好，但不确定图容量可靠性的计算过于复杂，而计算不确定图分布可靠性只需要获取其极小子图即可，为此，本章重新构建了一种基于流量和分布可靠性的关键边模型。针对此模型，首先提出了基于流量和分布可靠性的状态区间缓存算法 SCA_FDP（state caching algorithm with flow and distribution reliability），但是考虑到其在状态划分过程中存在初始状态划分空间大的缺点，本章提出了一种基于确定边过滤的优化算法 SCA_FEA（state caching algorithm with filter edge advanced），该算法使用割集边和悬挂边对于初始划分集合剪枝，从而减少初始状态划分集合大小。最后，通过实验分析，比较了基于流量和容量可靠性的模型与基于流量和分布可靠性的性能，同时比较了 SCA_FDP 算法和 SCA_FEA 算法对于不同图规模和不同稠密度再在空间复杂度和时间复杂度上的差别。

4.1 基于流量和分布可靠性的关键边模型

本文第三章研究的是基于流量和容量可靠性的关键边评估方法，然而计算不确定图的容量可靠性复杂度很高，该模型只能适应较小规模的图。因此本章提出一种基于流量（最大流）和分布可靠性（最可靠最大流分布可靠性）的不确定图关键边衡量模型。该模型针对不确定图中边移除（故障）后对流量和分布可靠性产生的相对损失，来评估关键边。该模型将流量作为衡量关键边的最重要因素，当移除边之后剩余不确定图的最大流一致时，比较剩余不确定图的（最可靠最大流）分布可靠性。假设 f，p_d 分别表示不确定图边移除之后剩余不确定图能够达到的最大流和最大流容量可靠性，K_f，K_{pd} 分别表示为最大流和容量可靠性的关键系数，那么模型中的指标关键程度可以表示为 $K_f>K_{pd}$。

因为计算不确定图的（最大流）容量可靠性，需要将不确定图所有满足最大流的子图都获取，然后计算所有满足最大流的子图概率之和。而相对于此，计算不确定图的分布可靠性就简单很多，同样也需要状态划分算法，但是根据极小子图理论[27]，只需要获取不确定图的极小子图即可，然后在极小子图上运行最大流算法获取（最可靠最大流）分布可靠性。

同样，根据流量和（最可靠最大流）分布可靠性，可以将不确定图的边分为三类，即边发生故障之后剩余不确定图和原不确定图进行对比：

（1）流量和分布可靠性都减少，作为 A 类边；

（2）流量不变，分布可靠性减少，作为 B 类边；

（3）流量和分布可靠性都不变，作为 C 类边。

根据这两个指标的关键程度 $K_f>K_{pd}$，易得出 ABC 三类边的关键程度依次减少。

在不确定图求解最可靠最大流分布的过程中，通过状态划分规则可以求解不确定图满足最大流的所有子图区间，根据边能够被满足最大流的子图的个数可对边进行分类。

根据不确定图子图的向量表示法以及向量的偏序关系，本章在获取的子图区间的基础上定义边在所有满足最大流子图的存在个数 S_i。

首先定义 s_{ij} 为边 e_i 在子图区间 j 的存在状况，根据偏序规则，其中 e_i=1 为在子图区间 C_j 中的所有子图都必须包含边 e_i；e_i=0 表示在子图区间 C_j 中的所有子图都不包含边 e_i；如果 e_i=x 则表示子图区间 C_j 中的子图有包含和不包含两种情况，定义 s_{ij} 如下：

$$s_{ij} = \begin{cases} 1 & (e_i = 1) \\ 0 & (e_i = x,0) \end{cases} \tag{4-1}$$

其中 i 表示边的序号，j 表示子图区间的序号。

对于不确定图中的每一条边 e_i 都有边存在个数 S_i，定义如下

$$S_i = \sum_{j=1}^{\tau} s_{ij} \tag{4-2}$$

其中 i 为边的序号，i∈{1….n}，n 为不确定图 G 边的个数，j 为子图区间的序号，j∈{1…τ}，τ 为使用状态划分后闭合区间的个数。

根据定义可知，S_i 有性质$0 \le S_i \le \tau$，其中τ为使用状态划分区间算法获得的最大流区间的个数。

根据边存在率 S_i 的定义可知，如果 $S_i=\tau$，那边边 e_i 存在于所有的满足最大流子图中，因此当 ei 发生故障时，所有满足最大流的子图都遭到破坏而不能满足最大流，从而相应的最大流容量可靠性也发生改变，因此当对于边 e_i，如果 $S_i=\tau$，那么 e_i 为 A 类边；如果 $0<S_i<\tau$，且边 e_i 存在于 G 的极小子图 g 中，那么边 e_i 发生故障之后，依然可以使用其他不包含该边的子图传递最大流，但是因为极小子图被破坏，所以最大流分布的可靠性下降，这样的边为 B 类边；同理，如果 $0\leqslant S_i<\tau$，且边 e_i 不存在于 G 的极小子图 g 中，那么边 e_i 发生故障之后，依然可以使用其他不包含该边的子图传递最大流，同时极小子图没有被破坏，所以最大流分布的可靠性不变，这样的边为 C 类边，不需要重新计算。

实例 4-1. 对于图 2-3 中的不确定图 G，其经过状态划分算法获取所有满足最大流的区间表示如表 3-1 所示，其中极小子图为 11011，根据公式 4-1，可知对于边 e_1，有 s_{11}=1，s_{12}=1，根据公式 4-2 可知，对于边 e_1 的边存在率 $S_1=s_{11}+s_{12}=2=\tau$，所以对于不确定图 G 而言，边 e_1 是一个 A 类边，同理可以知道边 e_1，e_2 和 e_5 为 A 类边；但是对于 e_4 而言，$S_4=1<2=\tau$，且 e_4 在极小子图上，所以 e_4 为 B 类边，移除 e_4 之后，依然可以使用子图 11101 满足最大流，但是因为极小子图 11011 被破坏，所以最可靠最大流分布可靠性下降；对于 e_3 而言，同样有 $S_3=1<2=\tau$，但是 e_3 不在极小子图 11011 上，因此 e_3 移除之后，极小子图没有被破坏，所以最可靠最大流分布可靠性也不变。

综上所述，根据边存在率 S_i，对边进行定性的分类如下：

1、$S_i=\tau$ ，那么 e_i 为 A 类边；

2、$0<S_i<\tau$且 e_i 存在于极小子图，那么 e_i 为 B 类边；

3、$0\leqslant S_i<\tau$且 e_i 不存在于极小子图，那么 e_i 为 C 类边；

根据以上的定性分析，可以得到如下的一系列的性质。

推论 4-1. 对于不确定图中的一条边 e_i，如果边 e_i 对应有 $0<S_i<\tau$ 且 e_i 存在于极小子图，那么移除 e_i，剩余不确定图依然可以达到最大流，但分布可靠性下降。

证明 4-1. 对于不确定图 G 的任意一条边 e_i，如果边 e_i 是 B 类边，即如果有 $0<S_i<\tau$ 且 e_i 存在于极小子图，下面将分两步证明（1）移除 B 边，不确定图最大流不变（2）移除 B 边，最可靠最大流分布可靠性下降。

（1）假设经过划分算保存下来的所有闭合子图空间集为$\{C_1，C_2，…，C_\tau\}$，满足 $C_i \cap C_j = \varnothing$，$C_1 \cup C_2 \cup \ldots \cup C_\tau = \mathrm{SET}$，其中τ 为满足最大流的子图区间的个数，因为 $0<S_i<\tau$，因此必存在一个子图$g \in \mathrm{SET}$，且 g 的 e_i 位置为 0，即移除 e_i 不会对子图 g 造成影响，又因为 g 能满足最大流，所以移除 e_i 不确定图的最大流不变。

（2）因为 e_i 存在于极小子图，所以移除 e_i 会使得不确定图的极小子图被破坏，那么不能满足原有的最可靠最大流分布可靠性。

推论 4-2. 对于不确定图中的一条边 e_i，如果边 e_i 对应有 $0\leqslant S_i<\tau$ 且 e_i 不存在于极小子图，那么移除边 e_i，不确定图的最大流，分布可靠性不发生改变。

证明 4-2. 对于不确定图 G 中的任意一条边 ei，如果 ei 是 C 类边，即如果有 $0<S_i<\tau$ 且 e_i 不存在于极小子图，下面分两步证明（1）移除 C 边，不确定图最大流不变（2）移除 C 边，最可靠最大流分布可靠性不变。

（1）同证明 4-1（1）。

（2）因为 e_i 不存在于极小子图，所以移除 e_i 不会使得不确定图的极小子图被破坏，那么依然能够满足原有的最可靠最大流分布可靠性。

4.2 基于流量和分布可靠性的关键边算法SCA_FDP

4.2.1 算法思想

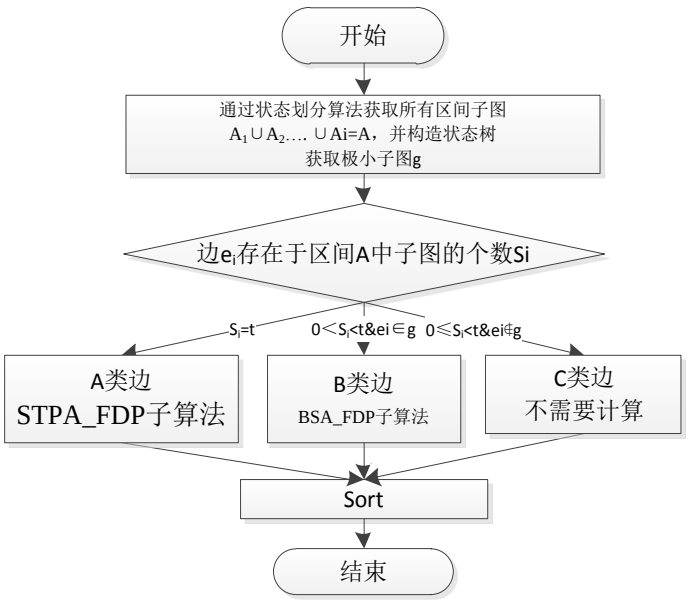


图 4-1 算法 SCA_FDP 流程图

本节在基于流量和分布可靠性模型的基础上提出了一种基于流量和分布可靠性的状态区间缓存算法 SCA_FDP，该算法的主要思想是根据模型中不同类型的边采用不同的子算法，以减少计算复杂度，对于 ABC 三类边做不同的处理。

基于以上步骤，SCA_FDP 算法可以表示为如图 4-1 步骤。

4.2.2 算法步骤

（I） A 类边的计算

对于不确定图 G 而言，求解（最可靠最大流）分布可靠性的方式目前普遍使用的是算法 SDBA[27]，SDBA 算法中使用到了状态划分算法，在状态划分的过程中，可以生成一个区间状态树，过程同算法 3-2。

对于 A 类边，本节提出一种基于割集状态树剪枝子算法 STPA_FDP（state tree pruning algorithm with flow and distribution reliability），该算法的主要是利用了割集中的边必定在最大流子图这一性质，通过割集中的边对于子图状态树进行剪枝，以达到减少搜索的目的。

该算法首先使用移除 A 边的剩余不确定图的最大子图获取割集，因为极小子图满足最大流，所以割集中的边必包含于极小子图，因此对于算法 3-2 生成的区间状态树中不包含割集中边的中间节点和叶子节点都必须舍去，对于依然保存的子图区间，进行进一步划分即可，直到获取极小子图。进一步划分操作如下：

（1）当区间的下界子图满足最大流，则整个区间子图都满足最大流，保留下界子图，并计算下界子图概率并与原有极小子图比较。

（2）当区间上界不满足最大流，则整个区间子图都不满足最大流，舍弃区间。

（3）区间下界不满足最大流，上界满足最大流，则需要二次划分区间。

通过这样的方法获取移除 A 边之后，剩余不确定图的极小子图，从而获取对应的最可靠最大流分布及其可靠性。

算法 4-2. A 类边的子算法 STPA_FDP

Input: uncertain graph G and it's source and sink, the breaked A class edge e, subgraph section tree T

Output: the distribution reliability p_d of uncertain graph (G-e)

1. get cut edge set by DINIC of MSG(G-e)， init $p_d \leftarrow 0$
2. init stack S and push all the left node of T (which pruned by cut edge set) into S
3. **While** S is not empty， **then** pop S as C
4. **If** lower(C) = F_{max} **then**
5. **If** p_d > the distribution reliability of lower(C) **then**
6. $p_d \leftarrow$ the distribution reliability of lower(C)
7. **End**
8. **End**

9. **If** lower(C) < F_{max} and upper(C) = F_{max} **then**
10. get sub-state of C with SDBA[27] as (C_1,C_2…C_n) and push (C_1,C_2…C_n) into Stack S
11. **End**
12. **End**
13. **Return** p_d

（II） B 类边的计算

对于 B 类边，由推论 4-1 可以知道，断掉 B 类边，最大流不会减少，此时的重点在于计算移除 B 边之后剩余不确定图的（最可靠最大流）分布可靠性。为了计算移除 B 边之后的分布可靠性，主要是在移除 B 边之后，找到剩余不确定图的新极小子图即可，换言之，就是要移除不能满足最大流子图，且不能存在重复的满足最大流的子图。本章提出子算法 BSA_FDP（B class edge search algorithm with flow and distribution reliability），用于获取 B 边移除之后，剩余不确定图的新极小子图。该算法步骤如下：

算法 4-3. B 类边的子算法 BSA_FDP

Input: The subgraph (that satisfy maximum flow) section of uncertain graph G and B class edge e_i

Output: The distribution reliability p_d of uncertain graph (G-ei)

1. init $p_d \leftarrow 0$
2. push the subgraph (that satisfy maximum flow) section of uncertain graph G into S
3. **While** S is not empty **then**
4. C ← get the top S and pop S
5. **If** the position of e_i in C is 1，**then** move out C
6. **If** the position of e_i position in C is 0，**then**
7. temp variable p ← reliability of lower(C)
8. If $p>p_d$ **then** $p_d \leftarrow p$
9. **End**
10. **If** the position of e_i in C is X，**then**
8. divide C to $C_{ei=1}$={e_1,e_2,…,e_{i-1},1,e_{i+1},…e_n} and $C_{ei=0}$= {e_1,e_2,…,e_{i-1},0, e_{i+1},…e_n}, then push $C_{ei=0}$ and $C_{ei=1}$ into S
9. **End**
10. **End**
11. **Return** p_d

BSA_FDP 算法的主要思想是在移除一个 B 类边之后，获取剩余不确定图的极小子图。根据推论 4-1 可以知道，B 边断掉之后，不确定图 G 任然能够满足最大流，所以剩余不确定图所有满足最大流的区间是移除 B 边之前满足最大流的区间的子集（移除包含该 B 边的子图）。

对于 C 类边的计算，根据推论 4-2，其断掉不会影响不确定图的最大流和（最可靠最大流）分布可靠性，因此不需要计算。直接使用原不确定图的最大流和分布可靠性即可。

基于以上处理方式，包含以上子算法的 SCA_FDP 算法的伪代码如下：

算法 4-4. 基于流量和容量可靠性的不确定图关键边算法 SCA_FDP

Input: uncertain graph G and it's source and sink

Output: sorted key edge set

1. Input uncertain Graph G and its source point and sink point
2. maxFlow, StateMtrix, StateTree ← algorithm SDBA[27]
3. ALLEdgeFlowDecrease[|E|] ← Get the max flow of the remain uncertain graph when every single edge being cut off by Dinic
4. sort ALLEdgeFlowDecrease[|E|] by max flow
5. Classify ABC edge by StateMtrix
6. i←0, j←1
7. **While** j < |E| **then**
8. **if** the max flow of the remain uncertain graph ($G-e_i$) equles the max flow of ($G-e_j$) **then**
9. check the class of e_i by step 5
10. **If** e_i is A class edge **then** use algorithm STPA_FDP to get p_d of ($G-e_i$)
11. **If** e_i is B class edge **then** use algorithm BSA_FDP to get p_d of ($G-e_i$)
12. **If** e_i is C class edge **then** p_d of ($G-e_i$) is same as G
13. **End**
14. i←j, j←j+1
15. **End**
16. Sort ALLEdgeFlowDecrease[|E|] by p_d
17. **Return** sorted key edge set

4.2.3 算法复杂度分析

基于流量和分布可靠性的状态区间缓存算法 SCA_FDP 首先需要获取所有边移除之后剩余不确定图的最大流，然后对计算的最大流进行排序，当前后两条边移除之后获得的最大流不相等时，根据模型定义，直接就可以比较出两条边的关键程度；当两条边移除之后最大流一致时，首先判断移除边的类型，然后根据三类边的不同性质，对于不同性质的边选择不同的算法获取移除该边之后剩余不确定图的容量可靠性，从而简化计算过程。

首先对于 A 类边，考虑最坏的情况，就是重复完全计算，该算法运算复杂度主要体现在 K 次运行 Dinic 算法，因此对于 A 边的复杂度为 $O(ak|V|^2|E|)$，其中 a 为不确定图 G 中 a 边的个数，对于 B 类边，需要遍历满足最大流的所有区间，其复杂度为 $O(b\tau)$，其

中 b 为不确定图 B 边的个数，τ 为满足最大流区间的个数，对于 C 边来说，因为不需要计算，只需要保留原有的最大流和容量可靠性，所以复杂度为 O(c)，其中 c 为不确定图 C 类边的个数。综上所述，SCA_FDP 的整体复杂度是 $O(|V|^2|E|^2)+O(ak|V|^2|E|)+ O(b\tau)+O(c)$，其中 a+b+c=|E|。

4.3 基于确定边过滤的优化算法SCA_FEA

4.3.1 算法基本思想

考虑到传统状态划分算法需要划分的状态空间较大，本节在原有计算方式的基础上提出一种基于确定边过滤的状态划分算法 SCA_FEA。该算法的主要思路是在使用状态划分算法之前，首先获取能够确定类别的边（包括割集中的边和悬挂边），从而减少状态划分的空间。

假设不确定图 G 的边集为 E，那么在传统的状态划分算法中，需要划分的状态区间中的子图个数有 $2^{|E|}$个，而在基于过滤集合的状态划分算法中，只要能够确定一条确定边（包括割集中的边或者悬挂边），都能够使状态划分区间中的子图减少一半，显然尽可能多的确定这些边，能够有效的简化算法复杂度。

SCA_FEA 算法的是相对于 SCA_FDP 的优化算法，减少了 SCA_FDP 算法状态划分的初始空间，因此 SCA_FEA 算法的主要目的是如何找到确定边。

目前求解网络中所有的割集已经被证明是一个 NP-Hard 问题，本节提供一种获取割集中的边算法，该方法不追求找到所有割集中的边，而是尽可能多的找到割集中的边，且满足找到的边一定是割集中的边。该方法的主要思想是首先用最大流最小割算法找到一组割，获取的最大流为全局最大流。然后，根据割集将图分割为两部分，并补全虚拟源点或虚拟汇点，连接割集边上的顶点到虚拟源点或者虚拟汇点作为一条边，并设置虚拟边的容量为一个大于原容量的值。接着，对构造的新图递归使用最大流最小割定理计算局部最大流，如果局部最大流等于全局最大流，则表示该最大流表示的割集也是所求割集，并将该子图继续分割，如果局部最大流大于全局，则舍弃该子图，直到没有可以计算的子图为止。那么获得的边即为所求部分割集中的边。

对于悬挂边，指的是只有入度或者只有出度的边（特殊的对于源点的入度边、汇点的出度边也属于悬挂边）。

4.3.2 算法步骤

（I） 割集中的边计算

目前求解图中所有割集的方法有割集矩阵算法，然而求解图的所有割集是一个 NP-Hard 问题，当图的规模越来越大，算法复杂度会上升很快，显然获取图的所有割集边是不合适的。本文使用一种递归的方式求解割集中的边，该方法的目的是尽可能多的获取割集中的边，而不能保证求得所有割集中的边。

在一般的图中，割集之间一般有两种形式存在，一种是独立的割集，另一种是关联割集。独立割集指两个割集之间没有重合的割边，而关联割表示，俩个割集中至少存在一条边公用。

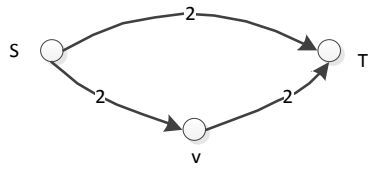


图 4-2 不确定图的最大子图

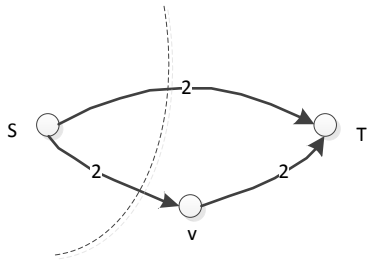


图 4-3 不确定图最大子图的割集表示

如图 4-2 为不确定图的最大子图 MSG(G），这个最大子图的其中一个割集如图 4-3 所示，那么 MSG(G）的的割集可以表示为{(ST，SV)(ST，VT)}，可以看出(ST，SV)(ST，VT)两个割集中包含相同的边，则表明两割集是关联割集。同理如果两个割集中不含有相同边，表示两割集为独立割集。

如果求解如图 4-2 中的割集，经过割集（ST，SV）划分，获得如下的两个分割图，如图 4-4 所以。

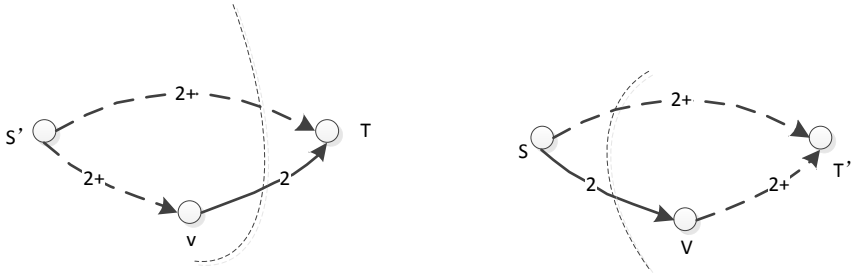


图 4-4 求得一个割集之后获得的两个子图

使用递归的方式求解所有子图中的局部最小割，直到没有分割图为止，返回所得的边，就是部分割集中的边。

一般情况下，图中也会存在一些悬挂边，这些边对于不确定图而言是最不关键的，通过悬挂边，也可以减少状态划分方法的初始划分区间，从而减少算法的复杂度。

基于以上分析，SCA_FEA 关于求解部分割集中边的算法流程图如图 4-5 所示。算法伪代码如下：

算法 4-5. 获取不确定图最大子图 MSG(G)割集中边

Input: the MSG(G) of uncertain graph G and it's source and sink
Output: some cut edge in MSG(G)

1. init queue Q, cut edge SET
2. get max flow F of MSG(G), push cut edge into SET and push divided subgraph into Q, the cut edge in the subgraph Assign c'=c+1
3. **while** Q is not empty **then** pop Q→g;
4. get partial max flow as f in graph g

5. **If** f > F then check the cut contains the original cut edges, if contains **then** adjustment else give up this subgraph
6. **If** f = F **then**
7. push Cut edge into SET
8. divid g into 2 subgraph as step(2) and push divided subgraph into Q
9. **End**
10. **End**
11. **Return** the edge in SET is some cut edge

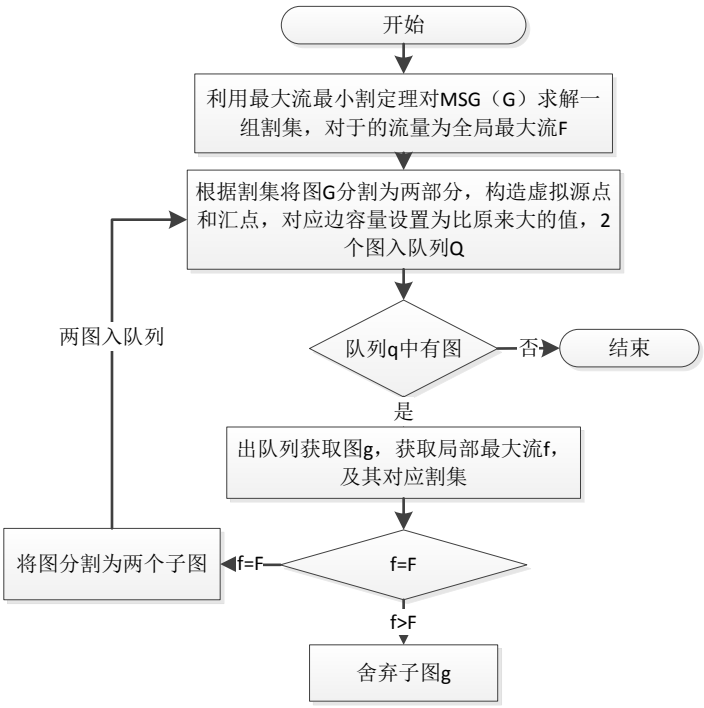


图 4-5 求解部分割集中的边的算法流程图

（II） 悬挂边的计算

悬挂边指的是只有入度或者只有出度的边（特殊的对于源点的入度边、汇点的出度边也是算作是悬挂边），在获取悬挂边的时候，只需要通过图中点的出入度判断即可。

如图 4-6 所示为一个比较特殊的不确定图的最大子图，可见点 v1 只有出度而没有入度，V2 只有入度而没有出度，那么 v1 便是单出度点，V2 是单入度点，又因为这些点都是中间节点，因此与这些点相连的边都不会出现在流分布当中，自然也不会出现在最大流分布当中，所以都可以在初始划分区间中置为 0，从而减少状态划分算法的划分空间，继而减少算法的复杂度。

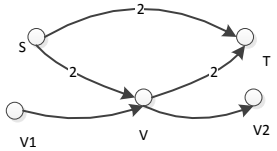


图 4-6 不确定图的最大子图

基于以上分析，计算悬挂边的算法流程如图 4-7 所示：

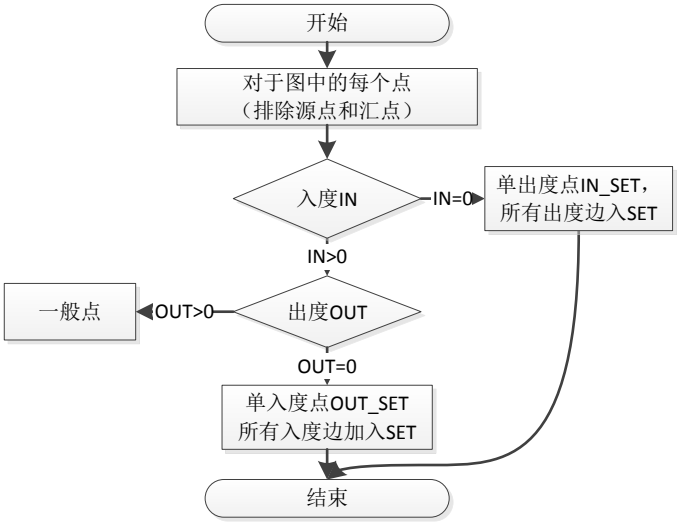


图 4-7 计算悬挂边流程图

算法伪代码如下：

算法 4-6. 获取不确定图最大子图 MSG(G)的悬挂边算法

Input: the MSG(G) of uncertain graph G and it's source and sink

Output: hanging edge SET

1. init IN_SET, OUT_SET, edge SET
2. **While** every single Node N in MSG(G) **then**
3. **If** Indegree = 0 **then** push N into IN_SET
4. **If** Outdegree = 0 **then** push N into OUT_SET
5. **End**
6. **for** node N in IN_SET, push the edge e into SET that connect to the node N which only have out flow
7. **for** node N in Out_SET, push the edge e into SET that connect to the node N which only have in folw
8. **Return** the edge in SET is the hanging edge

4.3.3 算法实现及分析

SCA_FEA 算法是在 SCA_FDP 的状态划分算法之前，获取了部分割集中的边和悬挂边，使得初始状态划分的空间缩小，从而算法减少复杂度。其中，使用割集中的边和悬挂边对于不确定图初始状态划分区间的设置如下：

（1）对于割集中的边，一定存在于满足最大流的子图中，将初始状态划分区间的上下界位置都设置为 1；

（2）对于悬挂边，不会存在于最可靠最大流分布中，将初始划分空间的上下界位置都设置为 0；

（3）对于处理后的初始区间，使用状态划分算法获取所有满足最大流的子图区间。

实例 4-2. 对于图 2-3 的不确定图，按照一般的情况下，以为不确定图 G 有 5 条边，那么初始的状态划分空间为 00000-11111，假设使用割集算法获取的割集中的边有 e1 和 e2，那么通过以上处理，获取的初始状态划分区间变成了 11000-11111，这样就使得原有的子图个数从 2^5 变成了 2^3，然后对于新的初始状态区间，再使用 SCA_FDP 算法获取关键边集合即可。

基于以上分析，使用该方法的流程图如图 4-8 所示：

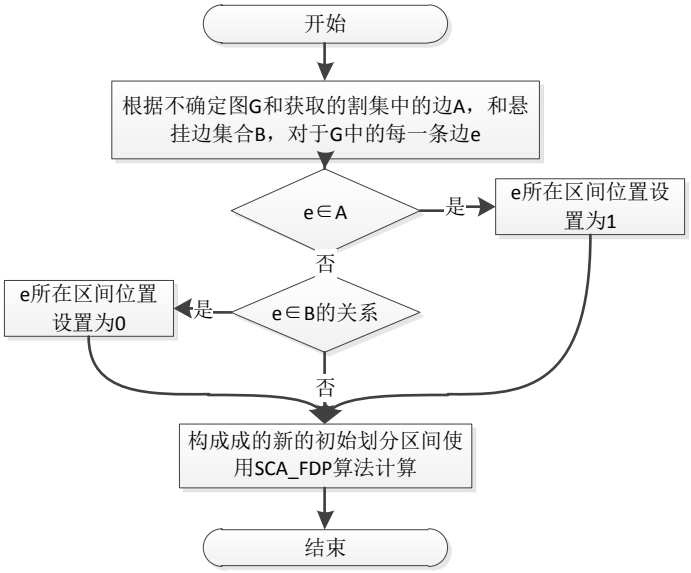


图 4-8 FEA 算法整体流程图

那么使用此方式的伪代码如下：

算法 4-7. 基于确定边过滤的算法 SCA_FEA

Input: uncertain graph G and it's source and sink

Output: sorted key edge set

1. init key edge set, The initial set of states $\{(0,0,\ldots,0_{|E|}),(1,1,\ldots,1_{|E|})\}$
2. SET_A ← get some key edge by algorithm 4-5
3. SET_B ← get hanging edge by algorithm 4-6
4. **for** $e_i \in G$ to states $\{(0,0,\ldots,0_{|E|}),(1,1,\ldots,1_{|E|})\}$
5. **If** $e_i \in$ SET_A **then**
6. set the low bound and upper bound of e_i position as 1 like $\{(0,0,\ldots,e_{i-1},1,e_{i+1}\ldots,0_{|E|}),(1,1,\ldots,e_{i-1},1,e_{i+1}\ldots,1_{|E|})\}$
7. **End**
8. **If** $e_i \in$ SET_B **then**
9. set the low bound and upper bound of e_i position is 0 like $\{(0,0,\ldots,e_{i-1},0,e_{i+1}\ldots,0_{|E|}),(1,1,\ldots,e_{i-1},0,e_{i+1}\ldots,1_{|E|})\}$
10. **End**
11. get new initial set of states and use SCA_FDP to get sorted key edge set

4.4 实验及分析

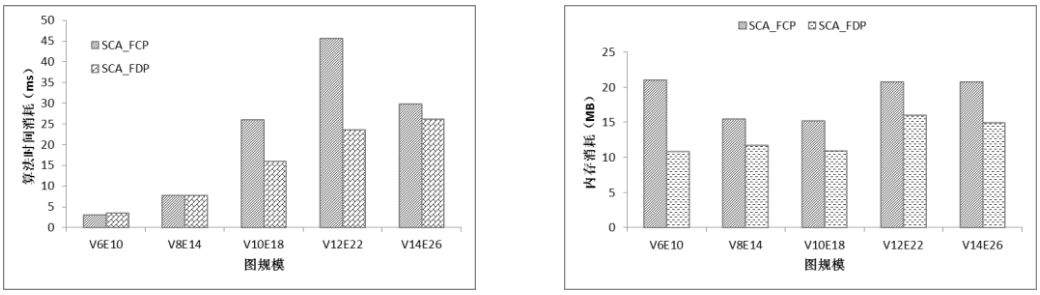
为了验证本章所提出算法的运行效率及分析影响算法性能各种因素，本节进行了一系列的实验，实验平台为一台 Intel Core 的 PC 机（CPU i7-3770，3.40GHz，内存 8GB，64 位 windows 7 操作系统），算法采用 C++在 VS2010 上实现。

4.4.1 实验数据

与第三章实验相同，本文采用了与文献[51]相同的数据集，使用 NETGEN 生成器产生 V_6E_{10}、V_8E_{14}、$V_{10}E_{18}$、$V_{12}E_{22}$、$V_{14}E_{26}$ 共 5 组不同图规模的二态有向图集合（本文实验数据集合大小为 5），其中 V_nE_m 表示有 n 个顶点、m 条边组成的图，图中边的容量与对应概率满足均匀分布。

4.4.2 实验结果与分析

实验 4-1. 不同图规模对于算法 SCA_FCP 和 SCA_FDP 性能的影响



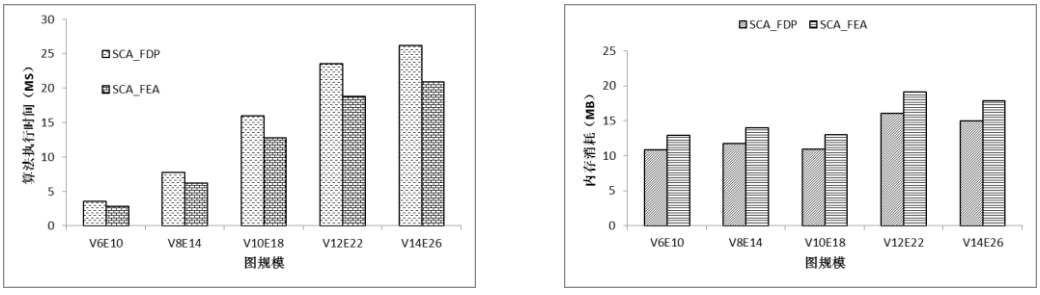
（a）时间消耗　　（b）内存消耗

图 4-9 不同图规模情况下 SCA_FCP 和 SCA_FDP 算法时间和内存消耗

如图 4-9 所示，SCA_FCP 算法是第三章中基于流量和容量可靠性的状态区间缓存算法，SCA_FDP 是本章基于流量和分布可靠性的状态区间缓存算法。如图 4-9（a）所示，在图规模较小的情况下，对于图 V_6E_{10}，算法 SCA_FDP 花费稍多的时间，但是随着图规模的增大，算法 SCA_FDP 的时间性能都优于算法 SCA_FCP。如果 4-9（b）所示，算法 SCA_FDP 在空间复杂度上，性能优于 SCA_FCP 算法，原因在于 SCA_FCP 在计算过程中需要获取并保存所有的不确定图满足最大流的子图，而 SCA_FDP 算法，只需要获取并比较满足最大流的区间下界子图即可。

综上所示，在不同规模的不确定图下，算法 SCA_FDP 的性能优于算法 SCA_FCP。

实验 4-2. 不同图规模对于算法 SCA_FDP 以及算法 SCA_FEA 性能的影响。



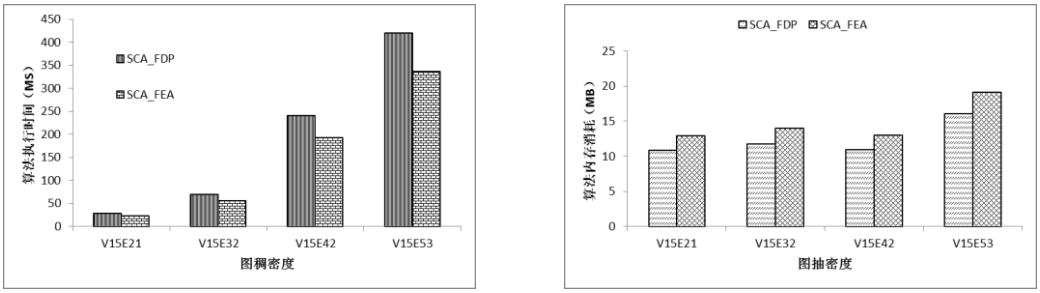
（a）时间消耗　　（b）内存消耗

图 4-10 算法 SCA_FDP 和 SCA_FEA 性能比较

如图 4-10（a）所示，在不同图规模下，SCA_FEA 相对于 SCA_FDP 在时间消耗上有了少量的降低。如图 4-10（b）所示，SCA_FEA 比算法 SCA_FDP 在内存消耗上稍多，但是随着图规模的增大，两个算法的内存消耗并没有增加很快，都在可以接受的范围之内。

实验 4-3. 不同图稠密度对于 SCA_FDP 和 SCA_FEA 性能的影响。

为更好的反映 SCA_FDP 算法和 SCA_FEA 算法的性能差异，使用 NETGEN 生成器生成 $V_{15}E_{21}$，$V_{15}E_{32}$，$V_{15}E_{42}$，$V_{15}E_{53}$ 四种不同稠密度的图，通过在不同图稠密度的情况下，比较两个算法在运行时间及内存消耗放方面的差异。试验结果如下：



（a）时间消耗　　（b）内存消耗

图 4-11 不同稠密度图算法 SCA_FDP 和 SCA_FEA 性能比较

如图 4-11 所示，相对于不同稠密度的图，算法 SCA_FEA 相较于 SCA_FDP，在时间消耗有一定的优势，在内存消耗上有少量增加。

4.5 本章小结

由于不确定图（最大流）容量可靠性的计算过于复杂，而计算不确定图（最可靠最大流）分布可靠性只需要获取不确定图的极小子图即可，为此，本章重新构建了一种基于流量和分布可靠性的关键边模型。针对此模型，首先提出了基于流量和分布可靠性的状态区间缓存算法 SCA_FDP。但是，考虑到该算法在状态划分过程中存在初始状态划分空间大的缺点，本章提出了一种基于确定边过滤的优化算法 SCA_FEA，该算法使用部分割集边和悬挂边对初始划分区间进行剪枝，从而减少初始状态划分区间的大小。最后，本章通过试验分析，比较了算法 SCA_FDP 和 SCA_FEA 在空间复杂度和时间复杂度上的差别。实验结果表明算法 SCA_FEA 相对于 SCA_FDP 在时间性能上都有了一定的提升，内存有少量的增加。同时相较于 SCA_FCP，SCA_FDP 算法在时间和空间复杂度上都有了优化。

然而，虽然本章提出的基于流量和分布可靠性的关键边模型能够有效的减少不确定图关键边获取的复杂度，但是，计算不确定图（最可靠最大流）分布可靠性依然是一个 NP-Hard 问题，这难以满足对计算速度要求很高的实时环境中。

第 5 章 基于流量和分布可靠性的关键边近似算法

因为计算不确定图的分布可靠性依然是一个 NP-Hard 问题，难以满足现实中对计算速度要求很高的实时环境中，为此，本章提出了一种基于流量和分布可靠性的关键边近似算法 KEAA_FDP（key edge approximation algorithm with flow and distribution reliability），实现（最可靠最大流）分布可靠性的快速计算，并给出了近似算法产生的关键边序列解和精确算法解的相似性度量，同时分析了近似算法的性能和波动性。

5.1 算法基本思想

本文第四章给出了基于流量和分布可靠性的关键边模型，虽然较基于流量和容量可靠性的模型有了较大的复杂度改进，但是对于现实中大规模网络，或者计算实时性要求比较高的系统，显然还是不适用的。经过分析发现在流量和分布可靠性模型中的 A 类边的计算，依然需要对于部分未知的状态区间进行二次划分，这将耗费很多时间。

本节针对基于流量和分布可靠性的关键边模型中 A 边移除之后需要二次划分的情况，提出一种对于模型中 A 类边计算的子算法，即基于剩余图最可靠简单路径优先增广的近似算法 ACES_MRSP（A class edge search base on most reliable simple path）。该子算法的主要思想是：首先，缓存原不确定图的最大子图在计算最可靠最大流分布的过程中的剩余图和简单路径，当模型中的 A 边发生故障之后，将所有包含该边的简单路径上的流量在剩余图中返还，调整剩余图中其他相关正向边和反向边上的容量，同时将“故障边”对应的正反向边在剩余图中移除。然后，在剩余图中查找所有满足连通源点 S 和汇点 T 的简单路径，选择一条“可靠性最大”的简单路径。在选择“可靠性最大的简单路径”时，将简单路径上“已使用未饱和边”概率设置为 1；而“未使用边”的概率用原容量对应的概率即可；“已饱和边”不可能出现在这样的简单路径上。具体的可以表示如下：设简单路径 sp 的的可靠性为 P(sp)，则：

$$P(\mathrm{sp}) = \prod_{e \in sp} p(e)$$

其中 p(e)为简单路径上边 e 对应的概率，那么简单路径上边 e 存在如下三种情况：

（1）对于当前流分布上边 e 为“已使用未饱和边”时，p(e)=1，即再使用该边时，不会使得流分布的可靠性减少；

（2）对于当前流分布没有使用到边 e 时，即 e 为“未使用边”，那么 p(e)为不确定图中边 e 存在的概率；

（3）当边 e 在当前流分布上为“饱和边”时，那么 e 不可能存在于这样的简单路径上。

最后，重复以上查找简单路径的步骤，直到没有从源点 S 到汇点 T 的简单路径即可。此时求得的便是移除该边之后，相对可靠的最大流分布。本章将使用该方法求得的流分布可靠性近似的作为衡量关键边的指标。

实例 5-1. 如图 5-1 所示不确定图和它的最可靠最大流分布，其中边 E_1，E_2，E_3，E_8，E_9 为它的 A 边，表 5-1 为不确定图 G 在求得最可靠最大流分布过程中获得的简单路径，可以知道原有不确定图的最大流 F_{max}=4。

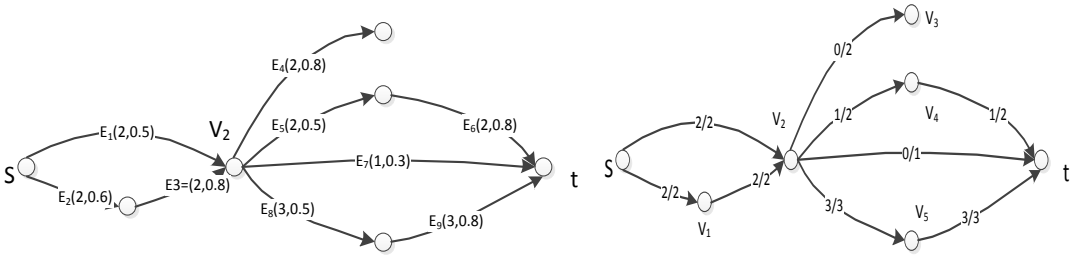


图 5-1 不确定图 G 与它的最可靠最大流分布

表 5-1 简单路径组合

编号	路径	容量
sp1	s-v_2-v_5-t	2
sp2	s-v_1-v_2-v_5-t	1
sp3	s-v_1-v_2-v_4-t	1

当边 v_2-v_5 发生故障（移除）之后，表 5-1 中的简单路径 sp_1 和 sp_2 都受到影响，那么移除 v_2-v_5 之后流分布及其对应的剩余图如图 5-2 所示。

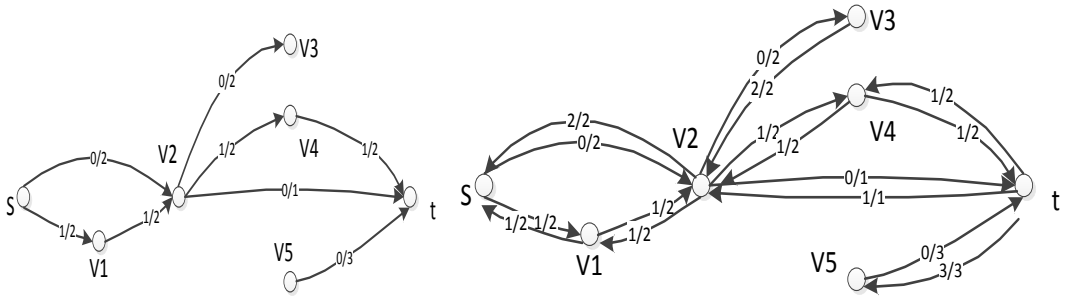


图 5-2 移除 v_2-v_5 后的图流分布及其对应的剩余图

此时在对应的剩余图中获取所有的简单路径，如表 5-2 所示。

表 5-2 移除边 v_2-v_5 之后的所有简单路径

编号	路径	瓶颈容量	可靠性
sp_1	s-v_2-v_4-t	1	0.5
sp_2	s-v_2-t	1	0.15
sp_3	s-v_1-v_2-v_4-t	1	1
sp_4	s-v_1-v_2-t	1	0.3

根据算法原理，可知简单路径 sp_1，sp_2，sp_4 都使用到了“未使用”边，而简单路径 sp_3 使用的都是“未饱和边”，而没有使用到“未使用边”。所以在这样的情况下，计算每条简单路径的可靠性如表 5-2 所示，选择一条可靠性最大的简单路径 sp_3。因为 sp_3 的

可靠性为 1，所以添加 sp_3 上的瓶颈流量之后，并不会使得不确定图的流分布可靠性下降，而流量却得到了增广。在添加这个简单路径之后，继续以上步骤，直到在剩余图中没有简单路径为止，因为在每次选择的都是可靠性最大的简单路径，因此该流分布可以认为是一个较可靠性的最大流分布。

5.2 算法实现与分析

根据上述分析，下面给出关于 A 边的基于剩余图最可靠简单路径优先增广的近似子算法 ACES_MRSP 的伪代码：

算法 5-1. 基于剩余图最可靠简单路径优先增广的近似算法 ACES_MRSP

Input: A class edge e_i in uncertain graph G

Output: Approximate results of reliability of the most reliable and maximum flow distribution

1. init original max flow distribution and all the simple paths
2. input A class edge e_i, remove the simple path which contains the edge e_i and adjust the flow in the original max flow distribution
3. **while** remain graph has simple path **then**
4. find all simple path on the remain graph, use the most reliable simple path
5. **End**
6. get the new max flow and it's distribution reliability

算法 ACES_MRSP 是针对基于流量和分布可靠性的关键边模型中，对于 A 类边移除之后，最大流分布可靠性的近似计算。该算法首先需要缓存原不确定图极小子图在计算最可靠最大流分布的过程中的剩余图和简单路径。然后，当需要计算模型中 A 边移除之后，剩余不确定图的（最可靠）最大流分布时，将所有包含该边的简单路径上的流量在剩余图中调整，同时将“故障边”在剩余图中移除。接着，在调整过的剩余图中查找所有满足能够连通源点 S 和汇点 T 的简单路径。选择一条“可靠性最大”的简单路径最后，重复以上查找简单路径的过程，直到剩余图中没有简单路径时，求得的最大流分布是相对可靠的最大流分布。

子算法 ACES_MRSP 的原理与 DINIC 类似，是在一个剩余图的基础上获取增广路径的，所以 ACES_MRSP 的最坏情况只有一个 DINIC 执行时间。

基于以上分析，算法 KEAA_FDP 的伪代码如下：

算法 5-2. 由 ACES_MRSP 组成的近似算法 KEAA_FDP

Input: uncertain graph G and it's source and sink

Output: sorted key edge set

1. Input uncertain Graph G and its source point and sink point
2. maxFlow, StateMtrix, StateTree ← algorithm SDBA[27]
3. ALLEdgeFlowDecrease[|E|] ← Get the max flow of the remain uncertain graph when

every single edge being cut off by Dinic

4. sort ALLEdgeFlowDecrease[|E|] by max flow

5. Classify ABC edge by StateMtrix

6. i←0, j←1

7. **While** j < |E| **then**

8. **if** the max flow of the remain uncertain graph ($G-e_i$) equles the max flow of ($G-e_j$) **then**

9. check the class of e_i by step 5

10. **If** e_i is A class edge **then** use algorithm ACES_MRSP to get p_d of ($G-e_i$)

11. **If** e_i is B class edge **then** use algorithm BSA_FDP to get p_d of ($G-e_i$)

12. **If** e_i is C class edge **then** p_d of ($G-e_i$) is same as G

13. **End**

14. i←j, j←j+1

15.**End**

16.Sort ALLEdgeFlowDecrease[|E|] by p_d

17.**Return** sorted key edge set

5.3 近似算法与精确算法相似度

因为不确定图的关键边算法获取的是不确定图边的关键度排序，这样不方便直接比较近似算法获取的结果的精确程度，为此，本小节设置了近似算法的相似度α。

首先，本章使用 w 表示算法获取的关键边排序精确结果与近似结果的平均欧几里得距离，距离 w 可以表示如下：

$$w = \frac{\sqrt{\sum_{e_i \in G}(O_{e_i} - O'_{e_i})^2}}{N} \tag{5-1}$$

如公式所示，，O_{e_i}表示为边 e_i 在精确算法中的位置，O'_{e_i}为边 e_i 使用近似算法求得的排序中的位置，N 表示为不确定图边的个数。

实例 5-2. 如表 5-3 所示，为一个有 6 条边的不确定图，通过精确算法和近似算法获取的不同的关键排序。那么根据相似度 w 的定义可以知道，w=0.33。

表 5-3. 近似算法与精确算法比对表

关 键 度 排序	精确算法求得的排序(数字代表边号)	近似算法求得的排序（数字代表边号)
1	1	2
2	2	1
3	3	4
4	4	3
5	5	5
6	6	6

经过分析可以知道在同等图规模的情况下，如果 w 越小，那么说明近似算法获取的不确定图边关键度排序越接近于精确解，也就是近似算法和精确算法的误差就越小。然而，w 与图规模是相关的，对于不同规模的不确定图情况下并不能说明。

为此本节提出不同图规模下近似算法的最差情况，然后把 w 和最差情况的比作为近似算法的与精确算法的误差，排除图大小的因素，用以比较不同图规模以及不同稠密度下，近似算法的波动性。根据分析可以知道，对于一段排序的相关性，最差情况可能存在两种情况：

（1）首末位置顺序颠倒，然后对剩下的串继续执行颠倒操作，直到结束。如对于一个精确排序为 1-2-3-4-5-6，与之相关的最差排序是 6-5-4-3-2-1。

（2）等步长排序颠倒，即每一个边位置的相差都是等步长的。如对于一个 1-2-3-4-5-6，与之相关的最差的排序是 4-5-6-1-2-3。

根据分析，对于猜测的两种情况，首末位置顺序颠倒才是最差的情况。下面给出证明过程。

根据相似性的定义，对于情况（1）首末倒置的情况有：

$$W_1 = \frac{1}{N}\sqrt{\sum_{i=0}^{N-1}(N-2i-1)^2} \tag{5-2}$$

对于情况（2），根据相似性定义有：

$$W_2 = \frac{1}{N}\sqrt{2 * (\mathrm{N}\%2) * (\mathrm{N}-\mathrm{N}\%2)^2} \tag{5-3}$$

对于情况（2）的相似度的表示，可以跟 N 为奇数和偶数分别表示如下：

$$W_2 = \begin{cases} \frac{1}{N}\sqrt{\frac{\mathrm{N}-1}{4}(\mathrm{N}+1)^2} & \mathrm{N}\text{为奇数} \\ \frac{1}{N}\sqrt{\frac{N^3}{4}} & N\text{为偶数} \end{cases} \tag{5-4}$$

如果要证明$W_1 \geq W_2$，则只需要分别证明当N为奇数和偶数的情况下成立即可。下面将分别使用数学归纳法证明。

I：当 N 为奇数时：

若要证明$W_1 \geq W_2$，只需证明$\frac{1}{N}\sqrt{\sum_{i=0}^{N-1}(N-2i-1)^2} \geq \frac{1}{N}\sqrt{\frac{\mathrm{N}-1}{4}(\mathrm{N}+1)^2}$，只需证明$\sum_{i=0}^{N-1}(N-2i-1)^2 \geq \frac{\mathrm{N}-1}{4}(\mathrm{N}+1)^2$即可。

（1）当 N= 1 时，有 0≥0 成立；

（2）假设当 N=K 时，且 K 为奇数，有$\sum_{i=0}^{K-1}(K-2i-1)^2 \geq \frac{\mathrm{K}-1}{4}(\mathrm{k}+1)^2$成立。

（3）则当N=K+2时（N仍然为奇数），若要证明$\sum_{i=0}^{K+1}(K+2-2i-1)^2 \geq \frac{\mathrm{K}+1}{4}(\mathrm{k}+3)^2$，经过化简只需要证明$\sum_{i=0}^{K-1}(K-2i-1)^2 + 2*(\mathrm{K}+1)^2 \geq \frac{\mathrm{K}-1}{4}(\mathrm{k}+1)^2 + \frac{(\mathrm{K}+1)^2}{2} + K^2 + 3\mathrm{K} + 2$，又因为当 N=K 时，有$\sum_{i=0}^{K-1}(K-2i-1)^2 \geq \frac{\mathrm{K}-1}{4}(\mathrm{k}+1)^2$成立，经过化

简只需要证明$2*(\mathrm{K}+1)^2 \geq \frac{(\mathrm{K}+1)^2}{2}+K^2+3\mathrm{K}+2$即可，即证明$\mathrm{K}^2-1 \geq 0$，显然对于 K=3,5,7….的奇数，是恒成立的。

所以当 N 为奇数时，有$W_1 \geq W_2$成立。

II：当 N 为偶数时：

若要证明$W_1 \geq W_2$，只需证明$\frac{1}{N}\sqrt{\sum_{i=0}^{N-1}(N-2i-1)^2} \geq \frac{1}{N}\sqrt{\frac{N^3}{4}}$，即只需要证明$\sum_{i=0}^{N-1}(N-2i-1)^2 \geq \frac{N^3}{4}$即可。

（1）当 N=2 时，有 2≥2 成立；

（2）假设当 N=K 时，且 K 为偶数，有$\sum_{i=0}^{K-1}(K-2i-1)^2 \geq \frac{K^3}{4}$成立。

（3）则当 N=K+2 时（N 仍然为偶数），若要证明$\sum_{i=0}^{K+1}(K+2-2i-1)^2 \geq \frac{1}{4}(\mathrm{k}+2)^2$，经过化简只需要证明$\sum_{i=0}^{K-1}(K-2i-1)^2+2*(\mathrm{K}+1)^2 \geq \frac{K^3}{4}+\frac{(6K^2+12\mathrm{K}+8)}{4}$，又因为当 N=K 时，有$\sum_{i=0}^{K-1}(K-2i-1)^2 \geq \frac{K^3}{4}$成立，所以只需要证明$2*(\mathrm{K}+1)^2 \geq \frac{(6K^2+12\mathrm{K}+8)}{4}$，经过化简只需要证明$K^2+2\mathrm{K} \geq 0$，显然对于 K=4,6,….的偶数，是恒成立的。

所以当 N 为偶数时，也有$W_1 \geq W_2$成立。

综上所述，当 N>1 时，都有$W_1 \geq W_2$，所以相对于一段排序，其相关性的下界是$\frac{1}{N}\sqrt{\sum_{i=0}^{N-1}(N-2i-1)^2}$，其中 N 为不确定图边的个数。

由于基于未饱和路径优先增广的近似算法主要是对与 SCA_FDP 算法相对于 A 类边需要状态区间二次划分的改进，在选择增广路径的过程中，会优先使用“未饱和边”来寻找一个较可靠的最可靠最大流分布。因此算法只会使得 A 边的排序造成影响，所以近似算法的最差情况可以表示为$\frac{1}{N_A}\sqrt{\sum_{i=0}^{N_A-1}(N_A-2i-1)^2}$，其中$N_A$表示的是基于流量和分布可靠性的不确定图关键边模型中 A 类边的个数。

根据以上分析，定义近似算法的相似性度量α如下：

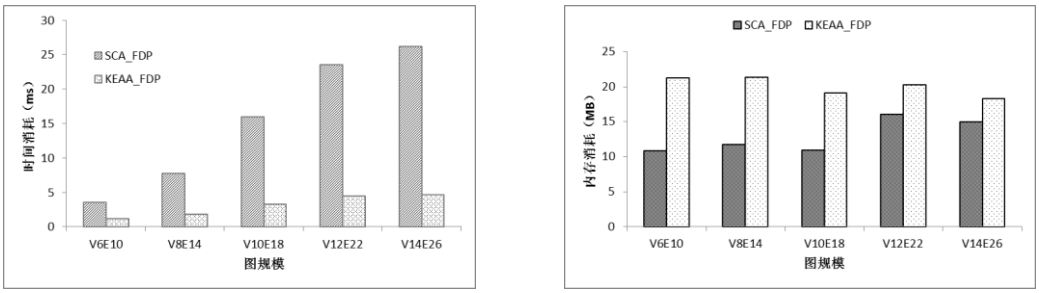
$$\alpha = 1-\frac{\mathrm{w}}{\mathrm{W}} = 1-\frac{\sqrt{\sum_{e_i \in G}\left(o_{e_i}-o'_{e_i}\right)^2}}{\mathrm{N_A}} \Big/ \frac{1}{N_A}\sqrt{\sum_{i=0}^{N_A-1}(N_A-2i-1)^2} \tag{5-5}$$

其中$\mathrm{N_A}$为模型之中 A 边的个数。相似性度量可以描述近似算法获取的不确定图关键边近似解与精确解的相似程度，且与不确定图的规模无关。对于大量图的分析，可以获取近似算法的波动性。

5.4 实验结果及分析

实验 5-1. 不同图规模对于算法 SCA_FDP 和算法 KEAA_FDP 性能的影响。

如图 5-3，SCA_FDP 是基于流量和分布可靠性的状态区间缓存算法；KEAA_FDP 是针对 SCA_FDP 中关于 A 类边的近似算法。如图 5-3（a）所示，在不同图规模下，算法 KEAA_FDP 相对于 SCA_FDP 在时间消耗上有了极大的降低。如图 5-3（b）所示，算法 SCA_FDP 和 KEAA_FDP 在内存消耗上随着图规模增大而升高，但是随着图规模的增大，算法的内存消耗并没有增加很快，都在可以接受的范围之内。

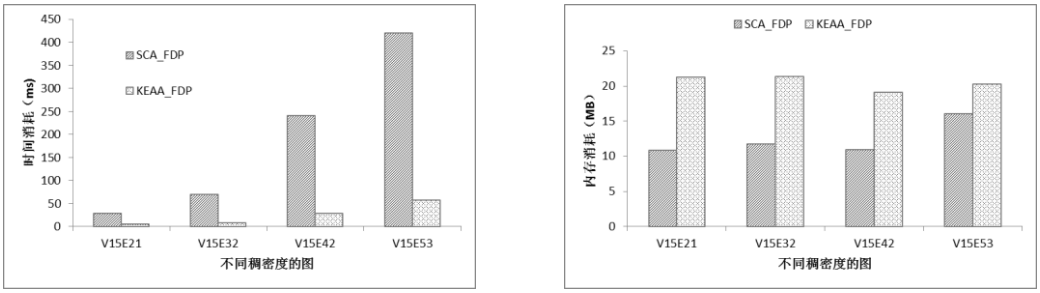


（a）时间消耗　　（b）内存消耗

图 5-3 SCA_FDP 和 KEAA_FDP 性能比较

实验 5-2. 不同图稠密度对于算法 SCA_FDP 和算法 KEAA_FDP 性能的影响。

为更好的反映 SCA_FDP 算法和 KEAA_FDP 算法的性能差异，使用 NETGEN 生成器生成 $V_{15}E_{21}$，$V_{15}E_{32}$，$V_{15}E_{42}$，$V_{15}E_{53}$ 四种不同稠密度的图，通过在不同图稠密度的情况下，比较两种算法在运行时间及内存消耗放方面的差异。试验结果如下：



（a）时间消耗　　（b）内存消耗

图 5-4 不同稠密度图 SCA_FDP 和 KEAA_FDP 性能比较

如图 5-4 所示，相对于不同稠密度的图，近似算法 SKEAA_FDP 比算法 SCA_FDP 在时间消耗上有很多优势，在内存消耗上有少量的增加，但可以接受。

实验 5-3. 近似算法 KEAA_FDP 波动性实验。

本实验对于近似算法 KEAA_FDP 验证其波动性。该实验的思路是首先使用精确算法 SCA_FDP 获取不同图规模的关键边排序，然后使用近似算法 KEAA_FDP 算法获取近似的关键度排序，对于不同的图，计算关键边序列的平均相对距离，继而获取相似性度量。关键边序列的相似性度量可以衡量关键边排序的相似程度。

本实验使用 V_6E_{10}、V_8E_{14}、$V_{10}E_{18}$、$V_{12}E_{22}$、$V_{14}E_{26}$ 共 5 组不同图规模的图数据，每个图规模下有 50 个不同的图，分别计算并获取平均相对距离，如下表 4-4 所示。

表 5-4 不同图规模下的相似性度量

图规模	平均相对距离	相似性度量
V6E10	0	100%

V8E14	0	100%
V10E18	0.015714	97.3271%
V12E22	0.018182	95.1845%
V14E26	0.023076	92.7771%

根据表 5-4 所示，平均相对距离是实验不同规模图数据精确解和近似解的相对欧几里得距离的平均值，近似算法和精确算法的误差为平均相对距离与最差情况的百分比，根据公式 5-5 获取的相似度如下图所示：

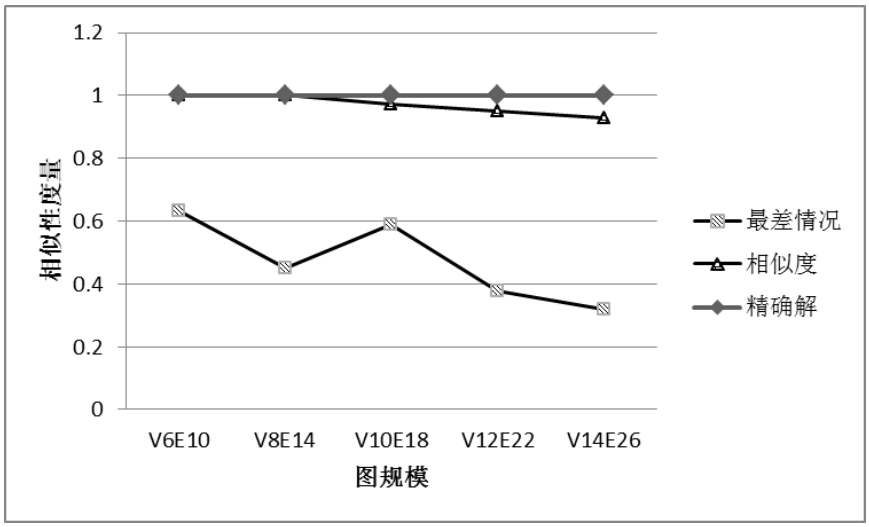


图 5-5 相似性度量比较

根据以上试验证明，在实验所做的图规模情况下，近似算法获取的近似解和精确解的相似度都在 95%以上，可以认为，在实验所做的图规模情况下，近似算法波动性不强。

5.5 本章小结

因为计算不确定图的分布可靠性依然是一个 NP-Hard 问题，难以满足现实中对计算速度要求很高的实时环境中，本章主要提出了一种基于流量和分布可靠新的关键边近似算法 KEAA_FDP，以实现（最大流）分布可靠性快速计算，并给出了近似算法产生的关键边序列解和精确算法解的相似性度量。最后，通过实验分析，近似算法具有较高的性能，且波动性不强，能够适应不同图规模和稠密度的图。

第 6 章 总结和展望

6.1 本文总结

本文主要是针对不确定图的关键边进行研究，首先对于问题相关概念进行了介绍，然后提出了基于流量和容量可靠性的不确定图关键边评估模型，基于此模型提出了基于流量和容量可靠性的状态区间缓存算法 SCA_FCP。而由于计算不确定图的容量可靠性复杂度较高，接着提出了基于流量和分布可靠性的关键边模型，并基于此模型提出了基于流量和分布可靠性的状态区间缓存算法 SCA_FDP，试验证明后者不管在时间复杂度还是空间复杂度上都有一定的优势。但是，因为计算不确定图分布可靠性依然是一个 NP-Hard 问题，所以本文接着提出另一种基于流量和分布可靠性的关键边近似算法，该算法避免了不确定图边移除之后的分布可靠性的重复计算，使得复杂度大大降低。最后，通过实验分析，比较各算法的时间和空间复杂度，同时验证了近似算法对于不同稠密度和图规模下的性能波动。

6.2 未来展望

本文的下一步研究主要包括：

1）本文主要研究的是二态不确定图关键边的研究，然而实际的系统中边的状态往往是多状态的，所以本文下一步研究将考虑在多状态不确定图下的关键边研究。

2）本文所考虑到的故障是单一的故障，即移除某一条边（单边故障），现实的情况是很复杂度，下一步研究将考虑多故障模型。

3）本文只考虑到如何识别关键边，对于关键边的改进可以提高整个网络的性能，下一步工作将考虑如果改进关键边以提高系统性能。

致谢

在这篇论文即将结束的时候，谨在此向在我硕士学习期间，对曾经关心和帮助我的各位老师、同学和朋友们致以衷心的感谢，感谢你们在这三年里，在学习和生活方面给于我的莫大帮助。

感谢我的导师张柏礼老师。回首这三年的时间，张老师在学习、科研、做人方面都耐心指导我，在我科研选题多次不理想情况下，给予我本文所讲述的题目。在修改毕业论文时候，监督并督促我的研究进度，张老师对凡事都认认真真的态度让我折服，对我一生影响都很大。

感谢实验室的吕建华老师。在张老师出国进修期间，吕老师在科研和生活中给我提出了很多的宝贵意见。

感谢实验室与我朝夕相处三年的小伙伴们。感谢虎哥、赵磊、翠翠和媛媛，你们将自己的知识与经验和我分享，同你们进行的每次探讨，都让我获益匪浅。

感谢我的家人。感谢他们多年给予我的鼎力支持和无私奉献，使我坚定信念客服困难，让我顺利完成课题的研究。

最后，再次感谢所有关心和帮助过我的人！

参考文献

[1] 任晓龙, 吕琳媛. 网络重要节点排序方法综述[J]. 科学通报, 2014, 59(13): 1175-1197.

[2] Newman M. Networks: an introduction[M]. OUP Oxford, 2010.

[3] Albert R, Jeong H, Barabási A L. Error and attack tolerance of complex networks[J]. nature, 2000, 406(6794): 378-382.

[4] Callaway D S, Newman M E J, Strogatz S H, et al. Network robustness and fragility: Percolation on random graphs[J]. Physical review letters, 2000, 85(25): 5468.

[5] Cohen R, Erez K, Ben-Avraham D, et al. Breakdown of the Internet under intentional attack[J]. Physical review letters, 2001, 86(16): 3682.

[6] Weng J, Lim E P, Jiang J, et al. Twitterrank: finding topic-sensitive influential twitterers[C]//Proceedings of the third ACM international conference on Web search and data mining. ACM, 2010: 261-270.

[7] Chen D B, Gao H, Lü L, et al. Identifying influential nodes in large-scale directed networks: the role of clustering[J]. PloS one, 2013, 8(10): e77455.

[8] Freeman L C. Centrality in social networks conceptual clarification[J]. Social networks, 1978, 1(3): 215-239.

[9] Katz L. A new status index derived from sociometric analysis[J]. Psychometrika, 1953, 18(1): 39-43.

[10] Estrada E, Rodriguez-Velazquez J A. Subgraph centrality in complex networks[J]. Physical Review E, 2005, 71(5): 056103.

[11] Dolev S, Elovici Y, Puzis R. Routing betweenness centrality[J]. Journal of the ACM (JACM), 2010, 57(4): 25.

[12] Kitsak M, Gallos L K, Havlin S, et al. Identification of influential spreaders in complex networks[J]. Nature physics, 2010, 6(11): 888-893.

[13] 李鹏翔, 任玉晴, 席酉民. 网络节点 (集) 重要性的一种度量指标 Ξ[J]. 2004.

[14] 陈勇, 胡爱群, 胡啸. 通信网中节点重要性的评价方法[J]. 通信学报, 2004, 25(8): 129-134.

[15] 谭跃进, 吴俊, 邓宏钟. 复杂网络中节点重要度评估的节点收缩方法[J]. 系统工程理论与实践, 2006, 26(11): 79-83.

[16] A.V.Goldberg. Recent developments in maximum flow algorithms(Invited Lecture). In: S.Arnborg,L.Tvansson eds. Pro of the Sixth Scandinavian Workshop on Algorithm Theory(LNCS 1004).Heidelber: Springer-Verlags1988,pp.1-10.

[17] Thomas H.Cormen Charles E.Leiserson, Ronald L.Rivest Clifford Stein. Introduction to Algorithms. Second Edition. MIT Press.2001.

[18] L.R.Ford, D.R.Fulkerson. Flows in networks. Princeton: Princeton University Press,1962.

[19] J.Edmonds, R.M.Karp. Theoretical improvements in algorithm efficiency for networks flow problems.1972(02).

[20] R.K.Ahuja,T.L.Magnanti,J.B.Orlin. Network Flows: Theory, Algorithms and Applications.2005:294-356.

[21] Clancy,D.O. Probability flows for reliability evaluation of multi-area power system interconnections. Electrical Power and Energy System,5,1983,pp.100-114.

[22] Lee SH. Reliability evaluation of a flow network. IEEE Transactions on Reliability 1980;29:24-6.

[23] Xue J. On multi-state system analysis. IEEE Transactions on Reliability 1985;34:329-37.

[24] M.O. Ball, Computing network reliability. Operations Research, vol.27, pp. 823–838, 1979.

[25] Zhao Peixin,Zhang Xin. A survey on reliability evaluation of stochastic flow networks in terms of minimal paths. Information Engineering and Computer Science 2009.

[26] Chen Y, Hu A Q, Hu J, et al. A method for finding the most vital node in communication networks[J]. High Technology Letters, 2004, 1(2): 573-575.

[27] 蔡伟，张柏礼，吕建华. 不确定图最可靠最大流算法研究[J]. 计算机学报，2012, 35(11): 2371-2380.

[28] 张柏礼，吕建华，生衍，等. 一种不确定图中最可靠最大流问题的解决方案[J]. 计算机学报, 2014, 10: 003.

[29] Lin JS, Jane CC, Yuan J. On reliability evaluation of a capacitated-flow network in terms of minimal path sets. Networks,vol.25. pp.131-138,1995.

[30] Lee SH. Reliability evaluation of a flow network. IEEE Transactions on Reliability 1980;29:24-6.

[31] Y.-K.Lin. System reliability evaluation for a multi-state supply chain network with failure nodes using minimal paths. IEEE.Trans.Reliability,vol,58,pp,34-40,2009.

[32] Y.-K.Lin. A Simple Algorithm for reliability evaluation of a stochastic-flow network with node failure. Computer & Operations Research 28(2001)1277-1285.

[33] J.E.Ramirez-Marquez, D.W.Coit. A monte-carlo simulation approach for approximating multi-state two terminals reliability. Reliability Engineering & System Safety.vol.87,pp.253-264,2005.

[34] P. Doulliez , E. Jamoulle. Transportation networks with random arc capacities. RAIRO, vol. 3, pp. 45–60, 1972.

[35] C. Alexopoulos. Note on state-space decomposition methods for analyzing stochastic flow networks," IEEE Trans. Reliability, vol. 44, pp.354–357, 1995.

[36] C.-C. Jane and Y.-W. Laih. A practical algorithm for computing multi-state two-terminal reliability. IEEE Trans.Reliability,vol.57,pp.295-302,2008.

[37] C.-C. Jane and Y.-W. Laih. Computing multi-state two-terminal reliability through critical arc states that interrupt demand. IEEE Trans.Reliability,vol.59.No.2,2010

[38] Yi-Kuei Lin, Cheng-Ta Yeh. Evaluation of optimal network reliability under components-assignments subject to a transmission budget. IEEE Trans. Reliability, 2010, 59: 539-550.

[39] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, et al. Introduction to algorithms. MIT Press, 2005

[40] L R Ford, D R Fulkerson. Maximum flow through a network. Canadian Journal of Math, 1956, 8(5): 399-404

[41] E A Dinic. Algorithm for solution of a problem of maximum flow in networks with power estimation. Soviet Math Dokl, 1970, 11(8): 1277-1280

[42] 江逸楠, 李瑞莹, 黄宁, 等. 网络可靠性评估方法综述[J]. 计算机科学, 2012, 39(5): 9-13.

[43] Zou Zhao-Nian, Li Jian-Zhong, Gao Hong, et al. Mining frequent subgraph patterns from uncertain graph data. IEEE Transaction on Knowledge and Data Engineering, 2010, 22(9): 1203-1218

[44] 张硕, 高宏, 李建中等. 不确定图数据库中高效查询处理. 计算机学报, 2009, 32(10): 2066-2079

[45] 张海杰, 姜守旭, 邹兆年. 不确定图上的高效 top-k 近邻查询处理算法. 计算机学报, 2011, 34(10): 1885-1896

[46] Potamias M, Bonchi F, Gionis A, et al. K-nearest neighbors in uncertain graphs. Proceedings of the VLDB, Singapore, 2010: 997-1008

[47] 袁野, 王国仁. 面向不确定图的概率可达性查询. 计算机学报, 2010, 33(8): 1378-1386

[48] Ye Yuan, Guoren Wang, Lei Chen, et al. Efficient Subgraph Similarity Search on Large Probabilistic Graph Databases. In Proc of VLDB, 2012: 800–811

[49] Khoussainova N, Balazinska M.Towards correcting input data errors probabilistically using integrity constraints. Proceedings of the MobiDE Workshop, Chicago, Illinois, USA, 2006: 43-50

[50] Han W S, Lee J, Pham M D. iGraph: A framework for comparisions of disk-based graph indexing techniques. Proceedings of the International Conference on Very Large Data Bases (VLDB), Singapore, 2010: 449-559

[51] Johnson D. Finding all the elementary circuits of a directed graph. SIAM Journal on Computing,1975,4(1):77–84.