基于流量和可靠性的不确定图关键边评估方法研究

作者名

摘 要: 网络中边的重要性评估是研究网络的一项重要内容，而现有网络中边的重要性评估方法片面强调最短路径（边介数），难以应用到与不确定图为背景的相关领域，本文构建了基于流量和可靠性（分布可靠性和容量可靠性）指标的不确定图关键边评估的数学模型，该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这一角度，对边的关键度进行综合评估。对于边移除之后，不确定图流量和可靠性相对变化的计算，本文首先设计了BASE算法作为基础，即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，才计算比较分布可靠性和容量可靠性，该算法作为正确性标杆；接着本文针对BASE算法重复计算复杂度较高的缺点，提出了一种基于状态划分树的增量算法ICA，该算法的主要思想是保存不确定图在状态划分过程中的子图区间，并构造子图区间树，通过子图区间将不确定图的边进行分类，当边移除之后，直接在子图树上按照一定的规则进行搜索即可获取依然能够满足最大流的子图，从而进一步获取对应分布可靠性和容量可靠性；最后本文通过实验，比较了两种算法的性能。实验结果表明ICA算法相对于BASE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。

关键词: 不确定图，关键边，最大流，分布可靠性，容量可靠性

# 引言

现实世界中, 网络形式的系统随处可见, 例如, 因特网、企业合作网络、产品供应链网络、客户关系网络等. 如何在复杂网络环境下, 保证网络的可靠性与抗毁性[ ], 以及网络中边的重要性评估[ ]是研究网络的重要内容. 由节点重要度评估找出那些重要的关键节点, 可以通过重点保护这些关键节点提高整个网络的可靠性. 因此, 对于网络中节点的重要度进行评估是一项有意义的工作.

评估网络中节点重要性的方法很多, 本质上都是源于图论[ ]. 最简单的方法是以节点的连接度(节点连接的边数) 作为节点重要度的衡量标准[ ], 认为与节点相连的边越多则该节点越重要. 这种评估方法具有片面性, 有些重要的”关键节点” 并不一定具有较大的连接度, 比如只有两条边相连的”桥节点”. 2002年Girvan和Newman在介数中心度的基础上, 提出了边介数[10](edge-betweenness)概念, 通过计算网络中边介数的大小来反映边对网络资小来反映边对网络资源的传输能力和控制能力的强弱, 边介数越大, 表明网络中任意节点对经过该条边的次数就越多，对网络资源的传输能力和控制能力就越强，在网络中所起到的作用也就越大, 因此, 边介数在一定程度上反映了边的重要程度; 文献[ ]提出了一种基于生成树数目的节点删除法, 定义最重要的节点为去掉该节点使得生成树数目最小的节点. 节点删除法的问题是如果多个节点的删除都使得网络不连通, 那么这些节点的重要度将是一致的, 从而使得评估结果不精确. 针对已有复杂网络节点重要性评估方法中片面强调节点的度而忽略了边对与之相连节点的支撑作用的缺陷, 熊金石等人提出了节点度和边介数共同作用下的评估数学模型[ ], 以体现边对其端节点的支撑作用.

然而在不确定图中，边存在发生故障的概率，在与不确定图相关的应用中，不仅要体现出边对网络传输能力的作用，同时也要体现出其对于可靠性支撑的作用，因此仅以边介数或节点度作为度量关键边的方法[11], 是具有片面性的，不适用于以不确定图为背景的相关领域。

在实际的应用场景中，不确定性是许多系统的固有特性[ ]，如电力传输网络中的元件、数据通信网络中的节点都存在着发生故障的概率，这种不确定性或是由于测量数据存在误差，或是应用本身包含概率特征而引入，当这种不确定性表达于图数据中，则形成不确定图[ ]。

根据最大流最小割定理，网络的最大流受限于网络中存在的最小割集，也就是说最小割集中的任意一条边断掉都会使整个网络可以传输的最大流量下降，这样的边就成了整个网络相对于流量的薄弱环节。同样的，不确定图中的边可以看做是从源点到汇点的信息传输通道，而关键边对于网络的信息传输可靠性起到支撑的作用，一旦发生故障，将对整个网络的信息传输产生巨大影响，甚至导致整个网络瘫痪，这样的边就构成了网络相对于可靠性的薄弱环节。为此，针对不确定图中边对于可靠性的重要度进行评估，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，对于薄弱环节的改进也可以显著的提高系统的可靠性。为此，本文构建了基于流量和可靠性指标的不确定图关键边评估的数学模型. 该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这一角度，对边的关键度进行综合评估。

需要介绍本文的两种方法

# 相关概念

本节对不确定图, 不确定图s-t流, 不确定图s-t流的可靠性等概念给出了相关的定义, 为问题和算法的描述提供方便.

**定义1.（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。



图1 不确定图G

例1: 图1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v1和v2, 边集为{E1, E2, E3, E4, E5}, 以边E4为例, 边上的容量c(E4)=1, 概率p(E4)=0.8, 也就是说E4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(E4)=0.2.

**定义2. (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’之后的不确定图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 不确定图G在移除边E4后的剩余不确定图G’

图2为不确定图G移除边E4之后的剩余不确定图G‘，G’中的顶点与原不确定图一致，但是边以被移除边上的流量概率对应被移除。

**定义3. （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3 不确定图G的子图g1 图4 不确定图G的子图g2

如图3和图4所示，g1和g2是图1中不确定图G的两个子图，根绝定义子图g2为不确定图G的最大子图，MSG（G）.

**定义4. （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。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。

例2：根绝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（g1）=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g2的子图概率为P(g2)=0.5\*0.5\*0.4\*0.8\*0.8=0.064.

**定义5.（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，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6.(不确定图最大流)** 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fmax为不确定图G的最大流.

例3：图1所示的不确定图G能够达到的最大流为2.

**定义8. （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 ，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。

**例4.** 如图5为图1所示不确定图G的三个能够满足最大流2的所有子图，那么不确定图相对于最大流的容量可靠性可以表示为P=p（g1）+ p（g2）+ p（g3）=0.176.



图5 不确定图G的三个能够满足最大流2的所有子图g1,g2,g3

**定义9. （不确定图流分布及其可靠性）**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 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。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。

**例5：**如图6为不确定图G的2个最大流分布F1和F2，分析可知，最大流分布F1被子图g2和g3所蕴含，则P（F1）=P(g2)+p(g3)=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(F1)=0.5\*0.5\*0.8\*0.8=0.16，结果一直。同理，对于分布F2，通过定义可知P（F2）=0.08.

 

图6 不确定图G的两个最大流分布F1和F2

**定义10. （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流分布, 其可靠性不小于该不确定图上的任一同等流分布. 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。

**例6.** 如图6中的两个分布是不确定图G的两个最大流分布，其中P（F1）>P(F2) ,且不存在一个其他的最大流分布F，使得P(F)>P(F1)，则F1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。

# 计算模型

不确定图能够达到最大流的大小，体现出网络的联通能力和传输能力，同时，分布可靠性体现出对应分布的稳定性和健壮性，这些性质对于不确定图的应用都是至关重要的，例如，网络的最大流是受限于网络中最在的最小割集的，也就是说最小割集中的任意一条边断掉都会使得真个网络可以传输的最大流下降，这样的边就成了整个网络相对于流量的关键环节，同理，这样的关键边也对于网络信息传输的可靠性起到支撑的作用，这样的环节一但发生故障，会使得整个网络的变的不稳定，甚至瘫痪。

基于此，本文构建了基于流量和可靠性指标的不确定图关键边评估的数学模型，该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这一角度，对边的关键度进行综合评估，本文指的可靠性为流分布可靠性，为增加区分度，同时考虑容量可靠性。

在研究该问题中，为简化模型，本文使用的是二态网络，即边的状态只有两种，且只考虑单故障情形，即只考虑一条边发生故障或断掉的情形，而不存在组合边发生故障的可能。其中边发生故障对不确定图造成的影响都是针对不确定图的最大流。本文的不确定图关键边衡量的模型用数学公式可以表示如下：

其中F，Pd，Pc分别为不确定图G在移除边e之后，剩余不确定图G‘能够达到相对的最大流，及其对应的流分布可靠性和容量可靠性。

在实际应用中，常在达到一定流量的情况下，考虑最可靠的流分布，因此本文在衡量关键边的时候，将优先比较能够达到的流量大小，即本文默认为流量因素是比可靠性更为关键的衡量因子。

# 计算方法

基于本文第三部分提出的不确定图关键边的衡量模型，计算不确定图边的关键程度主要体现在，当边移除之后，剩余不确定图能够达到的最大流，分布可靠性和容量可靠性的计算。本文首先设计了重复计算的基础算法（BASE algorithm），然后针对BASE算法重复计算复杂度高的缺点，提出一种基于状态划分树的增量算法ICA（Increment Cost Algorithm with State Interregion Tree）.

## 基于重复计算的基础算法

基于重复计算的基础算法（BASE algorithm），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，才计算比较分布可靠性和容量可靠性，该算法作为正确性标杆。

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1 BASE算法

输入：不确定图G，源点source，汇点sink

输出：根据关键程度排序好的keyEdgeSet

1. Input(G, source, sink);
2. Maxflow = Dinic(G, source, sink);
3. getALLEdgeFlowDecrease(key\_edge\_set, G, source, sink);
4. soetKeyEdge\_by\_flow(key\_edge\_set)
5. while flow(ei) == flow(ej)
6. reCompute(key\_edge\_set, g, source, sink, i);
7. soetKeyEdge(key\_edge\_set);
8. output(key\_edge\_set);

算法1首先要使用getALLEdgeFlowDecrease函数获取所有边发生故障之后能够满足的最大流，然后使用soetKeyEdge\_by\_flow函数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，而只有当两条边断掉获得流量相等的情况下，无法比较的情形，才会重复计算不确定图的分布可靠性和容量可靠性，使用分布可靠性和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的运行效率与重复计算的次数紧密相连，在算法的首先需要计算所有边发生故障之后的最大流，本文使用的计算最大流的算法是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)；然后当移除边之后流量相同的情况下，需要重复计算，本文使用的重复计算方式为基于状态划分的算法[ ]，其复杂度为O(k|V|2|E|)，其中K为划分过程中需要运行Dinic算法的次数。因此整个BASE算法的复杂度为O(km|V|2|E|)+ O(|V|2|E|2)，其中m为需要重复计算的次数，一般m的范围是0≤m≤|E|.

## 基于状态划分树的增量算法

虽然BASE算法相对于完全重复计算的方式减少了不少的重复计算，但是如果图规模较大的情况下，任然需要多次的重复计算，这显然对于大规模图或者高稠密图是不使用的。本节提出了一种基于状态划分树的增量算法ICA（Increment Cost Algorithm with State Interregion Tree），该算法的主要思想是保存不确定图在状态划分过程中的子图区间，并构造子图区间树，然后通过子图区间将不确定图的边进行分类，当边移除之后，直接在子图树上按照一定的规则进行搜索即可获取依然能够满足最大流的子图，从而进一步获取对应分布可靠性和容量可靠性

### 构造状态划分树

在不确定图G中通过状态划分规则可以生成一棵区间状态树。该树的生成规则如下：

1. 当区间处于已知满足最大流状态，则该区间作为一个叶子节点
2. 当区间



图7 完全状态划分树State-Tree

如图7所示，在完全状态划分树State-Tree的基础上，去掉不满足最大流的分支，则形成只满足最大流的状态划分树MaxFlow-State-Tree，MaxFlow-State-Tree的每一个分支的状态区间中的子图都满足最大流。



图8 只满足最大流的状态划分树MaxFlow-State-Tree

ICA算法首先根据状态划分规则获得的最大流子图区间将边进行分类, 然后对于不同类边设置不同的定量计算规则以减少复杂度，如图XXX为算法ICA的流程图。



图XX 算法ICA流程图

### 不确定图边的定性分类

根据状态划分规则[ ]可以求解不确定图满足最大流的所有子图区间，本文根据边能够被满足最大流的子图的个数对边进行分类。根据不确定图子图的向量表示法[23]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边在所有满足最大流子图的存在率Si。

首先定义sij为边ei在子图区间j的存在状况，根据偏序规则，其中ei=1为在子图区间j中的所有子图都必须包含边ei；ei=0表示在子图区间j中的所有子图都不包含边ei；如果ei=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sij如下：

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ei都有边存在率Si，定义如下

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。

根据定义我们可知，Si有如下性质：

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Si，对边进行定性的分类如下：

1、Si=*τ* ，那么ei为A类边；

2、0＜Si＜*τ*，那么ei为B类边；

3、Si=0，那么ei为C类边；

根据边存在率Si的定义可知，如果Si=*τ*，那么边ei存在于所有的满足最大流的子图中；如果Si=0，表示边ei，不存在任意一个满足最大流的子图中；如果0＜Si＜*τ*，表示边ei存在于部分，满足最大流的子图中。

根据以上的定性分析，可以得到如下的一系列的推论。

**性质1. 对于不确定图G中的一条边ei, 如果边ei是A类边，那么移除边ei会使得不确定图不能满足原有最大流。**

证明：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类边断掉，不确定图G任然可以达到最大流。下面将按照这两条分别证明；

（1）因为Ei是A类边，有Si=*τ*，根据定义边ei存在于所有的满足最大流的不确定子图中，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, …, *Cτ* },满足*Ci* ∩ *Cj =* ∅, *C*1 ∪ *C*2 ∪ … ∪ *Cτ* = *A*, 其中τ 为满足最大流的子图区间的个数, 因为Si=*τ* ，所以满足最大流的子图g一定有g∈A且有ei∈g，如果当ei发生故障之后，剩余不确定图G‘的任意子图g’都不包含ei, 即ei∉g’,则必有g∉A，因此对于ei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Si＜*τ*，则在A中必存在一个子图g，使得ei∉g，如果边ei发生故障，不确定图可以使用子图g传递流量，同时因为g∈A，即g能传递的最大流仍然为最大流，所以对于非A类边发生故障，不确定图G任然能够达到最大流；

的证。

**例7**：

**性质2. 对于不确定图中的一条边ei，如果边ei是B类边，那么移除ei，不确定图依然可以达到最大流，而最大流容量可靠性下降。**

证明：对于不确定图G的任意一条边ei，如果边ei是B类边，即如果有0<Si＜*τ* ，则ei移除之后，可以找不包含边ei的子图达到最大流，使得最大流不会下降，但是ei的断掉使得一部分能够达到最大流的子图遭到破坏，使得随机流网络的可靠性下降。

经过划分算保存下来的所有闭合子图空间集为{*C*1, *C*2, …, *Cτ* }, 满足*Ci* ∩ *Cj =* ∅, *C*1 ∪ *C*2 ∪ … ∪ *Cτ* = *A*, 其中τ 为满足最大流的子图区间的个数, 因为0<Si＜*τ* ，则对于任意的ei∈g子图，可以将子图的集合A分为g∈A1和g∉A2，其中满足A1∩A2=∅，A1∪A2=A。

在ei断掉之前，随机流网络的可靠性可以为A中的所有子图概率之和，可以表示为P(A)，而当ei断掉之后，集合A1中的子图不一定能够满足最大流，但是集合A2中的流量可以满可以满足最大流，因此保证了不确定图的连通性不变；同时ei断掉使得A1中至少有一个子图不能满足最大流，因此使得随机流网络的可靠性下降，此时即证明ei为不确定图的可靠性关键边。

的证。

例8：

**性质3. 对于不确定图中的一条边ei，如果ei是C类边，那么移除边ei，不确定图的最大流，分布可靠性和容量可靠性都不发生改变。**

证明：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Si=0，则ei断掉之后，原先满足最大流的子图任然能够满足最大流，使得不确定图的最大流不会改变，即连通性不会改变，随机流网络的可靠性性也不会改变，此时边ei为不确定图的非关键边。

想要证明Si=0时，ei是不确定图G的非关键边，只需要证明当ei断掉之后，不确定图的能达到的最大流和随机流网络可靠性不发生改变，即连通性和可靠性都不变即可。

经过划分算法保存下来的所有闭合子图空间集为{*C*1, *C*2, …, *Cτ* }, 满足*Ci* ∩ *Cj =* ∅, *C*1 ∪ *C*2 ∪ … ∪ *Cτ* = *A*, 其中τ 为满足最大流的子图区间的个数, 因为Si=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=（c1,c2…x…cn），其中ci表示为子图区间C的各条边对应的表示，i={1…i-1, i+…n}，则C可以分为两个子图区间（或子图）C1=（c1,c2…0…cn）和C2=（c1,c2…1…cn），当边ei发生故障之后，C2=C1，因为C1中的子图都能够满足最大流，故ei断掉之后C2中的子图也都能够满足最大流，所以不确定图的连通性和随机流可靠性都不会发生改变，此时ei为不确定图G的非关键边。

的证。

**例9**：

### 不确定图中边移除后定量计算

对边进行定性分类可以有效的减少重复计算的次数，对于不同的类别的边设置不同的计算方式是ICA算法的核心思想。

**（1）A 类边的计算**

对于A类边，由推理1可知，其断掉会使流量减少，即不能达到原有的最大流，因此此时原有最大流的网络可靠性和流分布的可靠性都为下降为0。

我们需要重新根据状态划分算法的极小子图理论对剩余不确定图，重新计算获得断掉之后的最大流及其对应的网络可靠性和分布的可靠性。



A类边定量计算正确性证明：

**（2）B类边的计算**

对于B类边，同样由推论1可以知道，断掉B类边，流量不会减少，此时的重点在于计算剩余的网络可靠性和流分布的可靠性。

对于网络的可靠性，如果子图区间中不包含该边，那么该区间中所有子图依然满足最大流；如果子图区间包含该边，那么判断该子图区间中的其他子图是否大于等于其他不包含该边的子图区间极小子图（可能存在多个）。如果满足，则表示其依然能够满足最大流，否则不能满足，网络的可靠性可以表示为这些断掉B类边任然能够满足最大流的子图概率之和。

对于流分布的可靠性，当一条B类边断掉之后，则判断这条边是否在极小子图上，如果不在极小子图上，则说明极小子图并没有被破坏，最大流分布没有被破坏，流分布的可靠性也不会减少；如果在极小子图上，则首先判断断掉该比边之后的区间下界子图是否满足最大流，如果满足则该区间生成新的下界子图，找到极小子图进行计算，如果不能满足最大流，则需要去除该子图区间，在剩余的区间中找到极小子图进行计算。

图xx 移除B类边计算随机流网络可靠性流程图

图xx 移除B类边，计算不确定图最可靠最大流分布的可靠性

**（3）C类边的计算**

对于C类边的计算，根据推论1，其断掉不会影响连通性，根据推论3，其断掉不会影响可靠性，也就是说C类边断掉不会对不确定图的状态照成任何的影响。

基于以上方式，ICA算法的伪代码如下：

**伪代码**

算法2 ICA算法

输入：不确定图G，源点source，汇点sink

输出：根据关键程度排序好的keyEdgeSet

1. Input(G, source, sink);
2. Maxflow, StateMtrix, StateTree= Dinic(G, source, sink);
3. getALLEdgeFlowDecrease(key\_edge\_set, G, source, sink);
4. soetKeyEdge\_by\_flow(key\_edge\_set)
5. classifyABC(StateMtrix, key\_edge\_set, AllEdgeFlowDecrease);
6. while flow(ei) == flow(ej)
7. if(ei is A Edge)
8. Search form StateTree arrording rule 1
9. If(ei is B Edge)
10. Search form StateTree arrording rule 2
11. if(ei is C Edge) Donot need compute
12. soetKeyEdge(key\_edge\_set);
13. output(key\_edge\_set)

### 算法复杂度分析

复杂度分析

# 试验及分析

试验部分

两个实验，

图规模大小的试验

图稠密度的试验

# 结束语

给本文做一个总结，研究了什么，做了什么

References: Reference

1. 作者. 题目. 刊名(全称), 出版年,卷号(期号):起始页码. [期刊] Text of Reference(当参考文献数<10时用) Text of Reference 1(当参考文献数≥10时用)
2. 作者. 书名. 版次(初版不写), 出版地(城市名): 出版者, 出版年. 起始页码(非必要项).[书籍]
3. 作者. 题目. In(中文用“见”): 整本文献的编者姓名ed(多编者用eds). 文集实际完整名称. 出版地(城市名): 出版者, 出版年. 起止页码.[会议录(论文集、论文汇编等)]
4. 著者. 题名. 学位, 学位授予单位, 出版年.[学位论文]
5. Author. Title. Technical Report, Report No., Publishing place (city name): Publisher, Year (in Chinese with English abstract).[科技报告]

附中文参考文献: 中文参考文献

[5] 著者.题名.科技报告,报告号,出版地(或单位所在地):出版者(或单位),出版年.