iPath算法

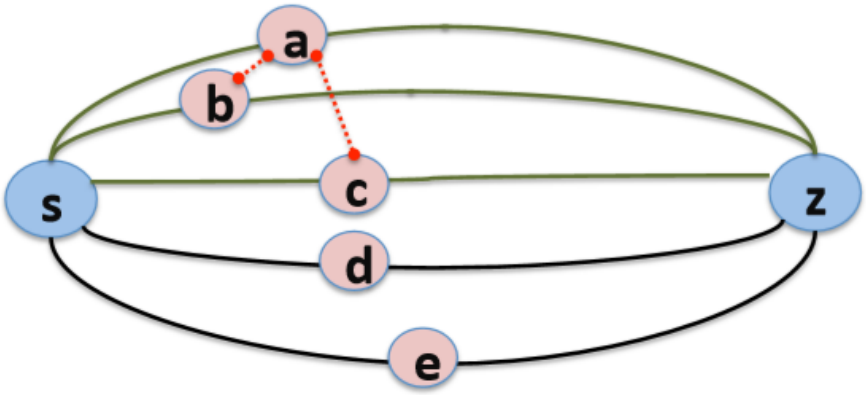
iPath算法
- 拜占庭容错通信
- 最优可靠径通信路径算法(iPath)
  - 模型简化
  - 算法流程
    - 符号说明
    - 简单证明
    - 算法实现
  - 简单测试

拜占庭容错通信

最优可靠径通信路径算法(iPath)

模型简化

俩跳网络(2-Hop network)，可自然推广到多跳网络。demo以俩跳为例。



算法流程

符号说明

P：全部顶点(s和z之间只有一个中继点，因此每个顶点也表示一个路径)

G：冲突图(conflict graph)中的顶点

O：P - G, i.e.不在冲突图中出现的其它顶点

δ：G的最小覆盖集的顶点数

S：$S \subseteq G$需要满足$|S| \leq 2\delta$并且S至少包括δ个faults，这个S的构建方法出现在reference paper 8中

C：$C = G - S$，C满足的性质是$\gamma^C = \delta$

C^*：$C^* \subseteq C$，且满足$|C^*| + |O| = 2(f - \delta) + 1$

简单证明

按上面的符号，有如下公式：

$$\begin{aligned}|C|+|O| =&|G|-|S|+|P|-|G|\\ =&|P|-|S|\\ \geq&|P|-2\delta\\ \geq&2f+1-2\delta\\ =&2(f-\delta)+1\end{aligned}$$

由于S中至少包括δ个faults，从而$C\cup O=P-S$最多含有$f-\delta$个faults，z收到$2(f-\delta)+1$个massage，只需要进行大多数投票就可以得出正确的信息。

且Lemma.2 证明了$2(f-\delta)+1$是下界，于是只需要选择C的子集C^*就可以了。

算法实现

下面按步说明算法的实现：

1. 创建数据结构 P、G、O
2. 求出G的最小覆盖集V_{min}，和δ
3. 在G中，对V_{min}和$G-V_{min}$做最大匹配，把$G-V_{min}$中被匹配的点的集合记作V_1
4. $S=V_1\cup V_{min}$
5. 计算C=G-S
6. 找出C^*,返回$C^*\cup O$

简单测试

我编了三个简单的测试，大概都只有10+个顶点，因为这个算法求解过程需要计算最小点覆盖和最大匹配问题，都是NP(C)问题，算法复杂度比较大，不知道大测试样例下会是什么样子。

```
% test1
% G_V_num = 7;
% G_O_num = 4;
% G_E =
[0,1,1,0,0,0,0;1,0,1,1,1,0,0;1,1,0,0,0,0,0;0,1,0,0,0,1,0;0,1,0,0,0,0,1;0,0,0,1,0
,0,1;0,0,0,0,1,1,0];
% test2
% G_V_num = 7;
% G_O_num = 4;
% G_E =
[0,0,1,0,0,0,0;0,0,1,0,0,0,0;1,1,0,1,1,0,0;0,0,1,0,0,0,0;0,0,1,0,0,1,1;0,0,0,0,1
,0,1;0,0,0,0,1,1,0];
% test3
G_V_num = 8;
G_O_num = 3;
G_E = [0,1,1,0,0,0,0,0;1,0,1,1,1,1,0,0;1,1,0,1,0,0,0,0;...
    0,1,1,0,0,1,0,0;0,1,0,0,0,1,0,0;0,1,0,1,1,0,1,1;...
    0,0,0,0,0,1,0,0;0,0,0,0,0,0,1,0,0];
```

test1是论文里面的样例。

简单解释下，G_V_num是conflict graph的顶点个数，G_O_num是没有冲突的路径数目。G_E是冲突图的邻接矩阵，索引从1开始，要求G_E是对称矩阵。