**修复算法测试报告**

# 性能测试

## 算法基础性能测试

### A\* 修复算法性能测试

1. 算法描述

原始A\* 算法，将模型所有节点进行优先级标记之后，将整个序列日志trace输入到A\* 算法中，对trace 所有修复方案进行遍历，寻找代价最小修复方案，遍历过程中使用代价（F=G+H）进行代价预估和剪枝叶。

1. 优缺点

优点：1.可输出trace修复最小代价值：min(cost)。

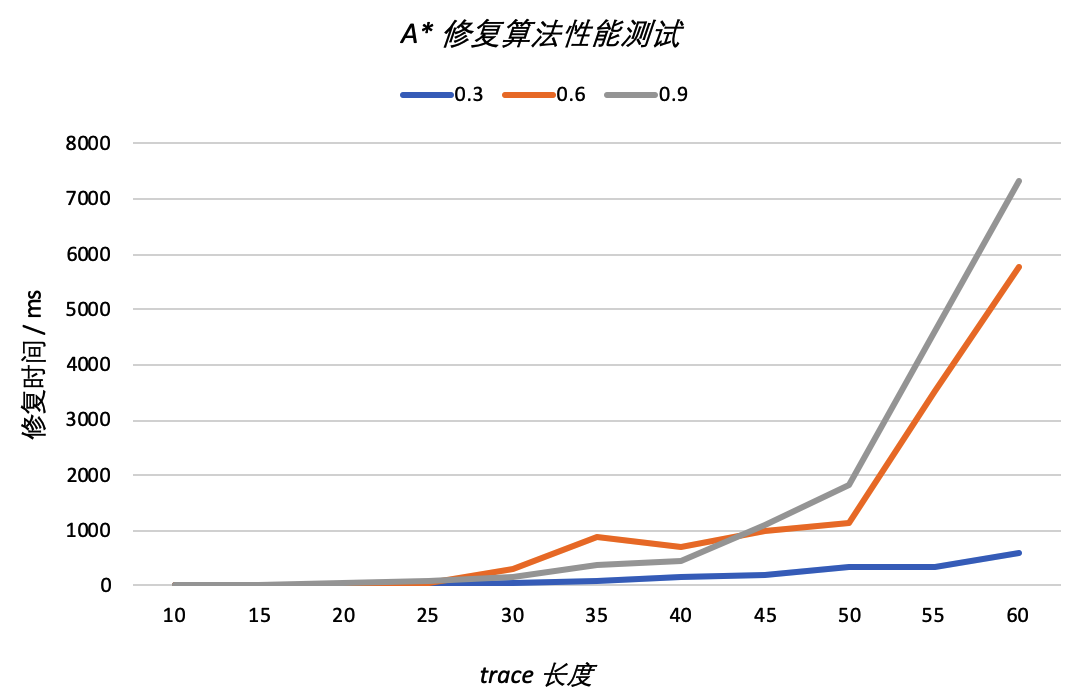
2.可获得trace中事件元素在修复过程中的移动方向和步长，并基于此信息作数。

据修复。

3.修复无误差。

缺点：速度较慢，需要对整体trace 做A\* 搜索树的遍历。

1. 测试结果



说明：横坐标：trace长度，纵坐标：修复时间（ms），三条曲线为trace不同的乱序率：

混乱30% 时间元素、混乱60% 时间元素和混乱90% 时间元素

### 2.基于结构的修复算法性能测试

1. 算法描述

针对一个需要修复的trace，首先对模型进行结构解析，trace序列中对应串行和分支结构的部分，可以根据模型信息直接得出修复结果，而对于无法通过模型结构信息得出的修复结构，例如：并行和循环等结构，通过原始的A\* 算法进行修复。

1. 优缺点说明：

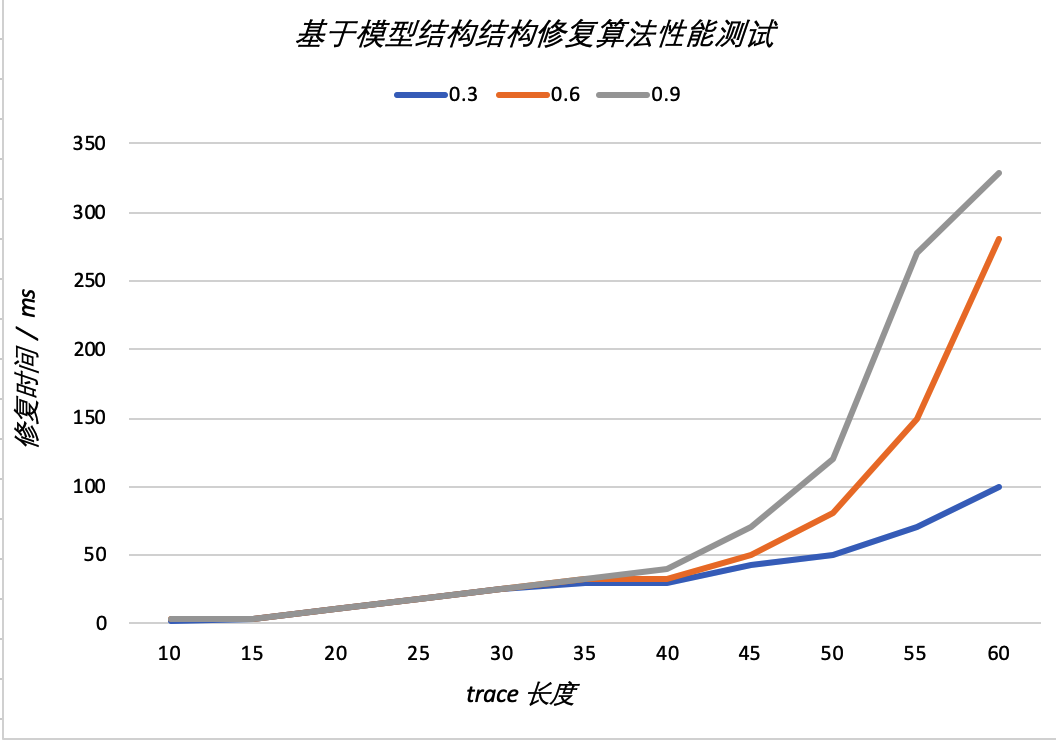
优点：1. 修复速度较快，仅针对部分trace片段进行A\* 修复。

2. 修复无误差。

缺点：1. 无法得知修复的最小代价信息：min(cost)。

2. 无法得知trace事件元素在修复过程中的移动方向和步长信息。

1. 测试结果：



说明：横坐标：trace长度，纵坐标：修复时间（ms），三条曲线为trace不同的乱序率：

混乱30% 时间元素、混乱60% 时间元素和混乱90% 时间元素。

### 3.最小预估代价修复算法性能测试

1. 算法描述

每次总是选择预估总代价最小的分支进行遍历，代价预估方式同A\* 算法相同，只贪心不回溯，在处理循环结构时，也使用预估代价选择最小预估代价的一个循环划分方式。

1. 优缺点说明

优点：1.速度最快，针对循环结构修复性能好。

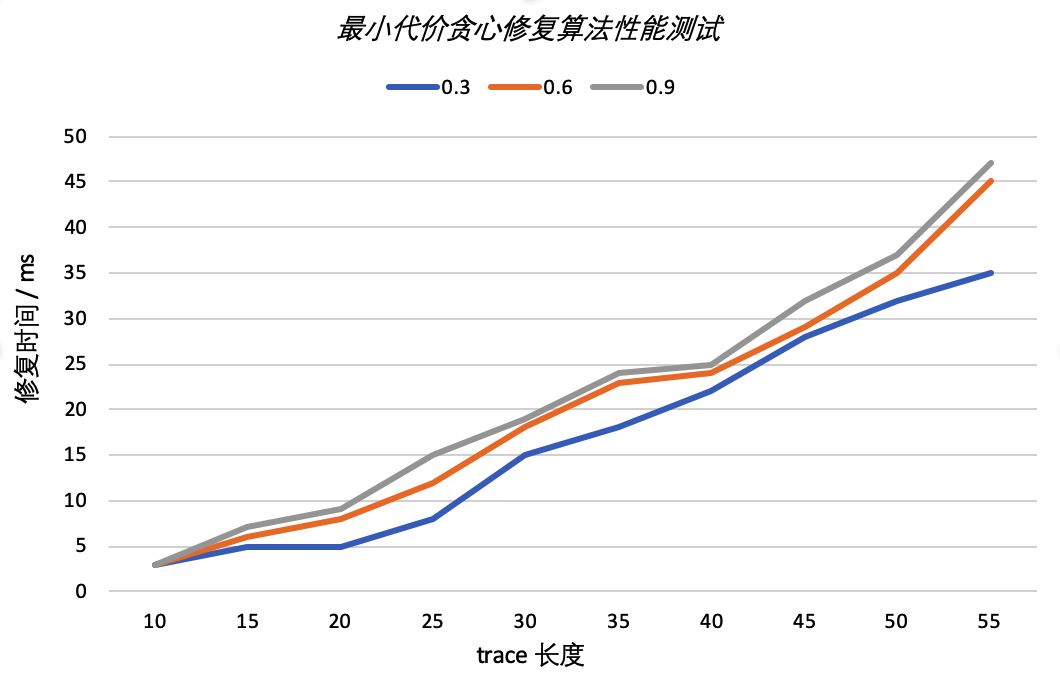
2.可输出trace修复最小代价值：min(cost)。

3.可获得trace中事件元素在修复过程中的移动方向和步长，并基于此信息作

数据修复。

缺点：1.修复存在误差，可能不是最优结果。

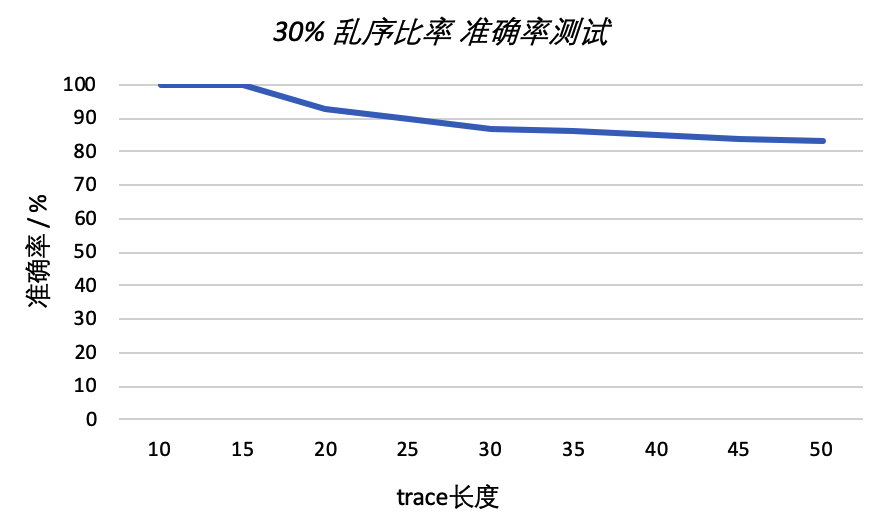
1. 性能测试结果



说明：横坐标：trace长度，纵坐标：修复时间（ms），三条曲线为trace不同的乱序率：

混乱30% 时间元素、混乱60% 时间元素和混乱90% 时间元素。

1. 准确率测试结果



问题： 正确率偏低，后期。考虑在贪心时，除了top1的划分方案，还把topN 内的划分方案同时加入到候选方案中，在性能和准确率中选取一个平衡点。

## 循环结构算法性能测试

在处理循环结构时，需要将多次循环进行切分，然后将trace中的事件元素划分到最合适的一次循环中，从而转化循环结构为非循环结构。

在常见的循环结构中，往往会包含串行、并行、分支和子循环结构，针对每一次循环，串行和并行结构时能够按照就近原则划分到最合适的一次循环中，但是分支和子循环结构，无法按照就近原则进行划分，所以循环的处理主要是针对循环中分支和循环子结构的划分处理。

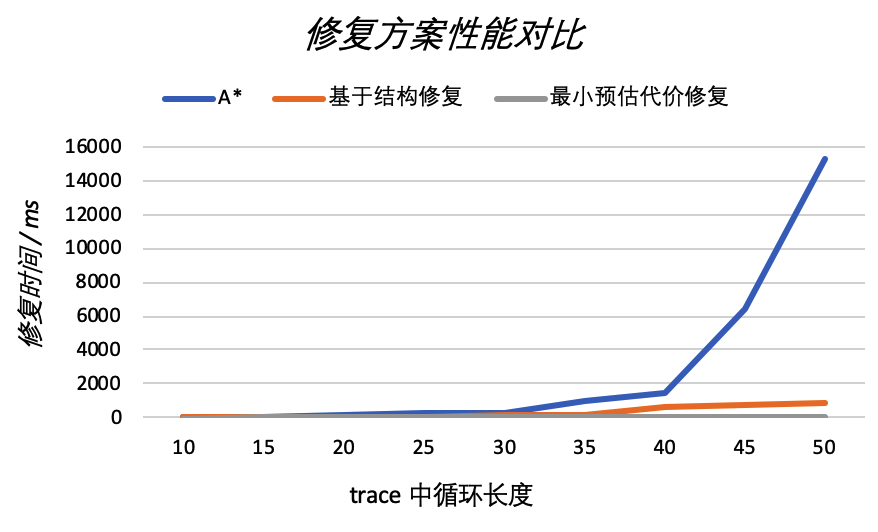
处理方法：

1. 组合划分方法

针对每一个分支和子循环结构，尝试所有的归属划分方案。

1. 最小预估代价方法

基于A\* 算法中的代价预估，对方法1中的组合划分方法中每一种划分方案，计算预估总代价，选取最小的一个作为划分方案。

性能对比结果：  


说明：本次试验中，trace错乱比率固定为30%，每个循环固定循环次数为3.

其中A\* 和基于结构修复的方案采用组合划分方式，最小预估代价，采用第二种最小预估代价的方案。