数据挖掘 GSP

数据挖掘中的GSP算法（Hash^2 GSP）。本文由周可人，李强，曾钢，张伍召完成。你可以给我们发送邮件： `kerenzhou@outlook.com` 。

假如你想了解我们工作的整个框架，请阅读 `report.pdf` 。

特点：

摘要：

1. 支持min gap和max gap
2. 带有 状态剪枝 和 自适应hash参数 的Hash-tree
3. Time-list匹配算法

具体：

1. 有两种输入数据格式，你可以更改 `-file_type` 参数， `0` 代表通用输入数据， `1` 代表和spmf项目一致的格式。我们在 `data` 目录下提供了一些测试数据。

   common input data:

   例子是 `100.txt` ， `seq.txt and gen.data` ，格式如下：

   sequence_id number_of_items item_id item_id …

   sequence_id number_of_items item_id item_id …

   sequence_id number_of_items item_id item_id …

   每行代表一个item_set，其中包含了不同的item

   spmf input data:

   例子是 `BMS1_spmf.txt` ， `kosarak10k.txt` ， `kosarak25k.txt` ， `and small.txt` ，格式如下：

   item_id -1 item_id item_id -1 … -1 item_id -2

item_id -1 item_id item_id -1 … -1 item_id -2

item_id -1 item_id item_id -1 … -1 item_id -2

item_set被 `-1` 分隔， `-2` 代表sequence的结束。

2. 在我们的源码中有一个 数据生成器 ，但是这些特点并没有完全的拓展。现在，数据生成器可以支持 高斯分布 和 均匀分布 。在你使用之前首先用更改文件中的参数，并且需要重新编译。

Usage:

```
./gsp -i [file_name] -t [support: float] -sequNUM [unsigned int32] -min
[unsigned int32] -max [unsigned int32] -eventNUM [unsigned int32] -
file_type [0:common, 1:spmf]
```

举个例子, 如果你需要使用在 `../data/` 目录下的 `100.txt` ，你需要键入如下命令：

```
./gsp -i ../data/100.txt -t 0.5 -sequNUM 100 -min 2 -max 4 -eventNUM 100 -
file_type 0
```

优点:

1. 在数据比较大的时候，本文提出的算法比原GSP算法有10倍的提升。并且比Prefixspan算法更快。实验数据如下：

kosarak10k.txt 10000 sequences

algorithm	dataset	support	time(s)
hash^2 gsp	kosarak10k	0.05	0.133
gsp	kosarak10k	0.05	0.235
prefix	kosarak10k	0.05	0.232
hash^2 gsp	kosarak10k	0.04	0.099

gsp	kosarak10k	0.04	0.382
prefix	kosarak10k	0.04	0.23
hash^2 gsp	kosarak10k	0.03	0.15
gsp	kosarak10k	0.03	0.454
prefix	kosarak10k	0.03	0.24
hash^2 gsp	kosarak10k	0.02	0.207
gsp	kosarak10k	0.02	1.217
prefix	kosarak10k	0.02	0.272
hash^2 gsp	kosarak10k	0.01	0.484
gsp	kosarak10k	0.01	4.373
prefix	kosarak10k	0.01	0.372

kosarak25k.txt 25000 sequences

algorithm	**dataset**	**support**	**time(s)**
hash^2 gsp	kosarak25k	0.05	0.299
gsp	kosarak25k	0.05	0.436
prefix	kosarak25k	0.05	0.345
hash^2 gsp	kosarak25k	0.04	0.233
gsp	kosarak25k	0.04	0.631
prefix	kosarak25k	0.04	0.42
hash^2 gsp	kosarak25k	0.03	0.323
gsp	kosarak25k	0.03	0.921
prefix	kosarak25k	0.03	0.425
hash^2 gsp	kosarak25k	0.02	0.339
gsp	kosarak25k	0.02	2.22
prefix	kosarak25k	0.02	0.591
hash^2 gsp	kosarak25k	0.01	0.611

hash^2 gsp	kosarak25k	0.01	0.611
gsp	kosarak25k	0.01	8.95
prefix	kosarak25k	0.01	0.7

BMS1_spmf60k.txt 60000 sequences

algorithm	**dataset**	**support**	**time(s)**
hash^2 gsp	BMS1_spmf60k	0.05	0.093
gsp	BMS1_spmf60k	0.05	0.188
prefix	BMS1_spmf60k	0.05	0.188
hash^2 gsp	BMS1_spmf60k	0.04	0.078
gsp	BMS1_spmf60k	0.04	0.796
prefix	BMS1_spmf60k	0.04	0.191
hash^2 gsp	BMS1_spmf60k	0.03	0.109
gsp	BMS1_spmf60k	0.03	0.797
prefix	BMS1_spmf60k	0.03	0.235
hash^2 gsp	BMS1_spmf60k	0.02	0.124
gsp	BMS1_spmf60k	0.02	2.58
prefix	BMS1_spmf60k	0.02	0.312
hash^2 gsp	BMS1_spmf60k	0.01	0.453
gsp	BMS1_spmf60k	0.01	22.9
prefix	BMS1_spmf60k	0.01	0.485

进展：

12/9/2014:

1. 自适的hash-tree分枝的方法
2. 多线程技术

3. 改进的数据生成器