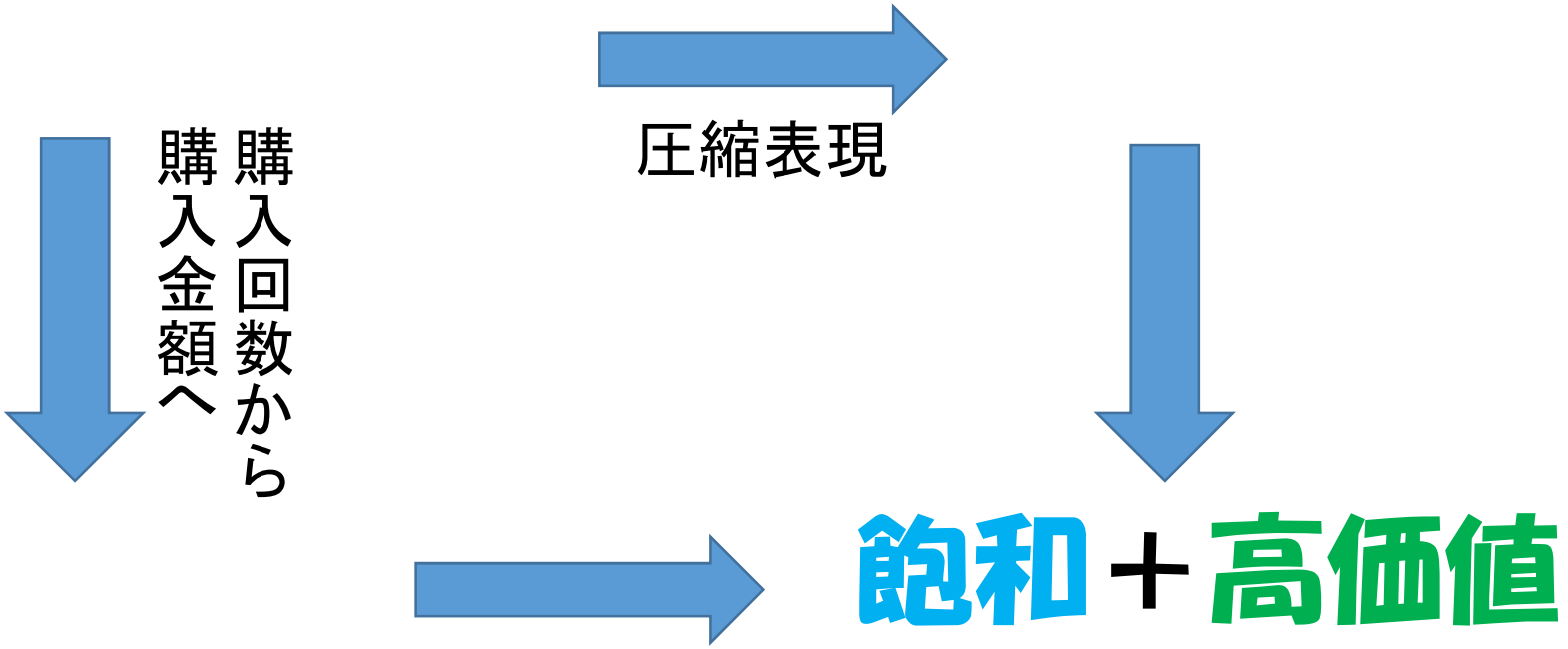
GPGPUによる
飽和高価値
アイテム集合マイニング

尾崎 研究室

5410010

栗山 裕介

GPGPUによる
飽和高価値
アイテム集合マイニング



アイテム集合マイニング

Database: D

Tid	商品名（価格）
Tr1	A(300),B(500), C(100),D(200), E(15000)
Tr2	A(500), C(150), E(10000)
Tr3	C(200),E(5000), F(400)
:	:
TrN	

Tr1, Tr2に出現 ⟶ A,C: 2

パターン　頻度 (購入回数)

頻出パターン

・ユーザーが決めた購入回数以上買われたパターン。

購入回数が最小購入回数以上出現するパターンを見つけたい！

最小購入回数を20回以上とした場合

A:35,　B:35, C:42, D:29,

AB:35, AC:26, BD:25, CD:29,

ACD:26, BCD:25, ・・・

飽和アイテム集合マイニング

Tid	商品名（価格）
Tr_1	A(300),B(500), C(100),D(200), E(10000)
Tr_2	A(500),C(100), E(10000)
Tr_3	C(200),E(5000), F(400)
:	:
Tr_N	

飽和アイテム集合
頻出パターンの圧縮表現

頻出パターン

A:35,　B:35, C:42, D:29,
AB:35, AC:26, BD:25, CD:29,
ACD:26, BCD:25, ・・・

頻度が同じものを
グループ化(同地類)

頻出パターンを経由せずに
パターンを得られる

飽和アイテム集合

導出可能

極大限のみ
求める

A:35, B:35, AB:35

AC:26,　ACD:26

BD:27,　BCD:27

D:29,　CD:29

C:42

高価値アイテム集合マイニング

Tid	商品名（価格）
Tr_1	A(300),B(500), C(100),D(200), E(15000)
Tr_2	A(500), C(150), E(10000)
Tr_3	C(200), E(5000), F(400)
⋮	⋮
Tr_N	

Tr_1,Tr_2,Tr_3に出現

CE ： 3， 30450

パターン　支持度　価値

高価値アイテムパターン

・ユーザーが決めた最小金額以上出現したパターン

購入価格が最小金額以上のパターンを求めたい！

最小購入金額を10000円以上とした場合の高価値パターン

A:35,150000　B:35,180000

C:42,215000　D:29,185500

E:5,1000000 AB:35,330000

CD:29,340000　BCE:5,1300000 ・・・

得られる高価値パターンの数が膨大になる

飽和・高価値アイテム集合マイニング

Tid	商品名（価格）
Tr_1	A(300),B(500), C(100),D(200), E(15000)
Tr_2	A(500),C(100), E(10000)
Tr_3	C(200),E(5000), F(400)
:	:
Tr_N	

飽和･高価値パターン
高価値パターンの圧縮表現

高価値パターン

A:35,150000　B:35,180000
C:42,215000　D:29,185500
E:5,1000000　AB:35,330000
CD:29,340000　BCE:5,1300000・・・

頻度が同じものを
グループ化(同値類)

高価値パターンを
経由せずに導出

飽和･高価値アイテム集合

A:35,150000 B:35,180000 AB:35,330000

D:29,185500　CD:29,340000

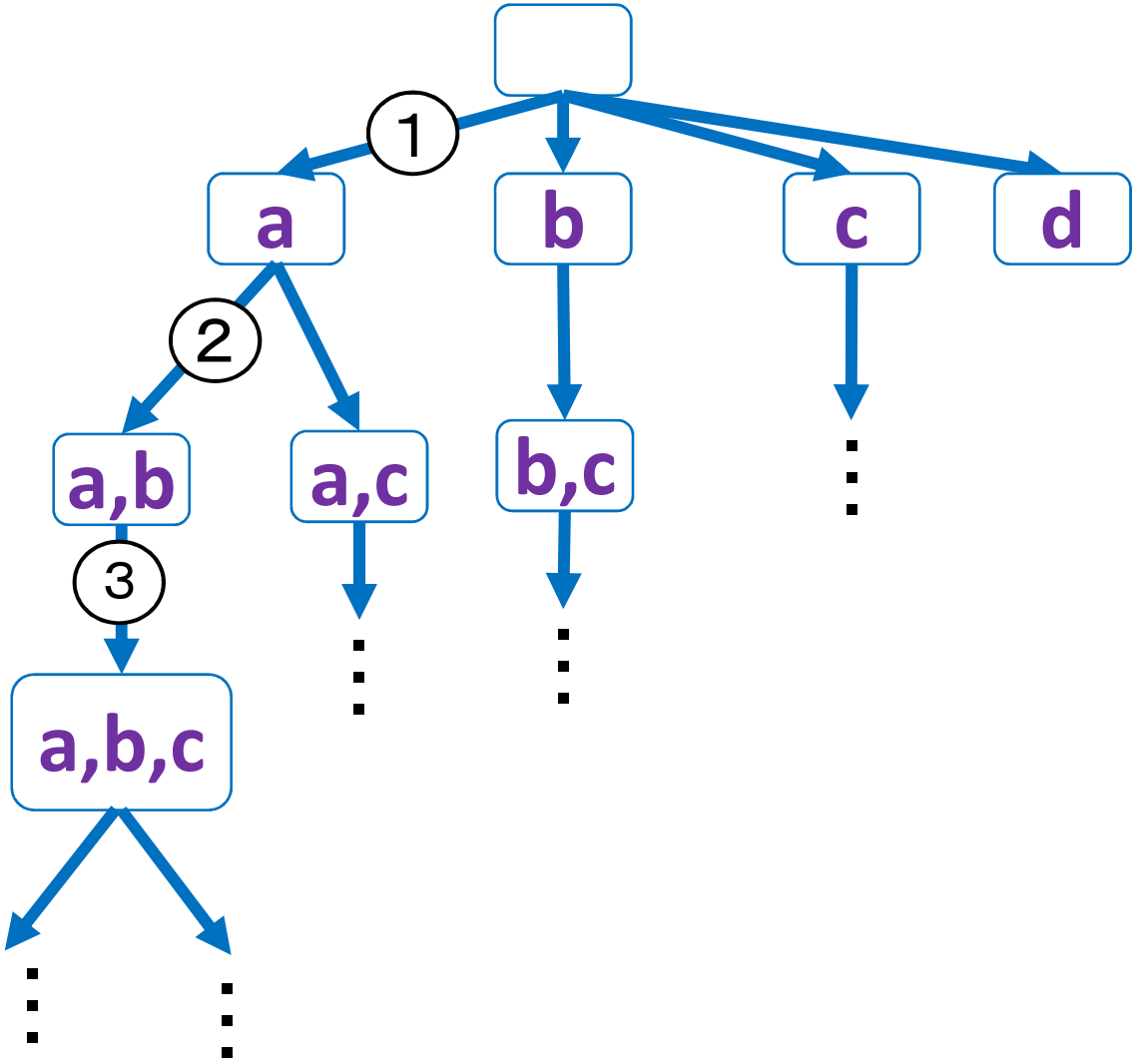
E:5,1000000　BCE:1,1300000

C:42,215000

極大限のみ
求める

アルゴリズム CHUD [Wu2011]

集合列挙木の縦型探索

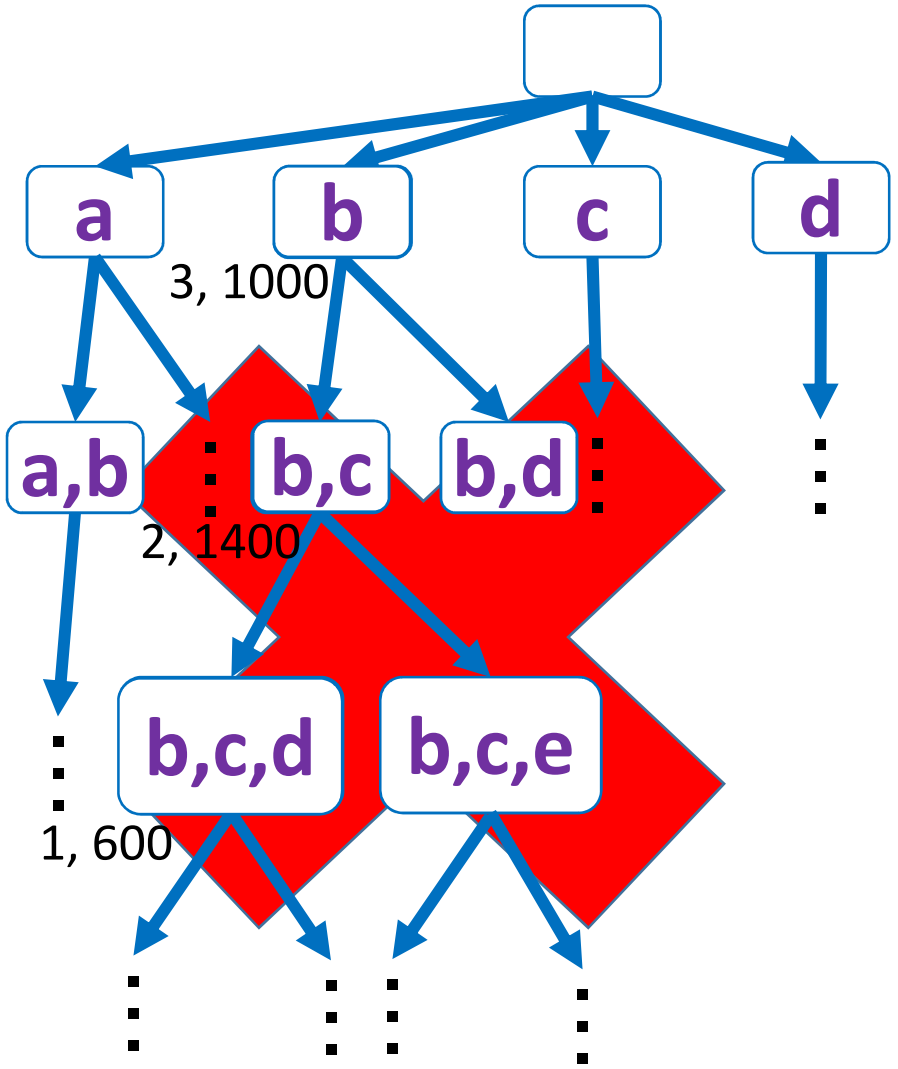


- 上界値による枝刈り
- 左の枝刈り
- 右の枝刈り

上界値による枝刈り

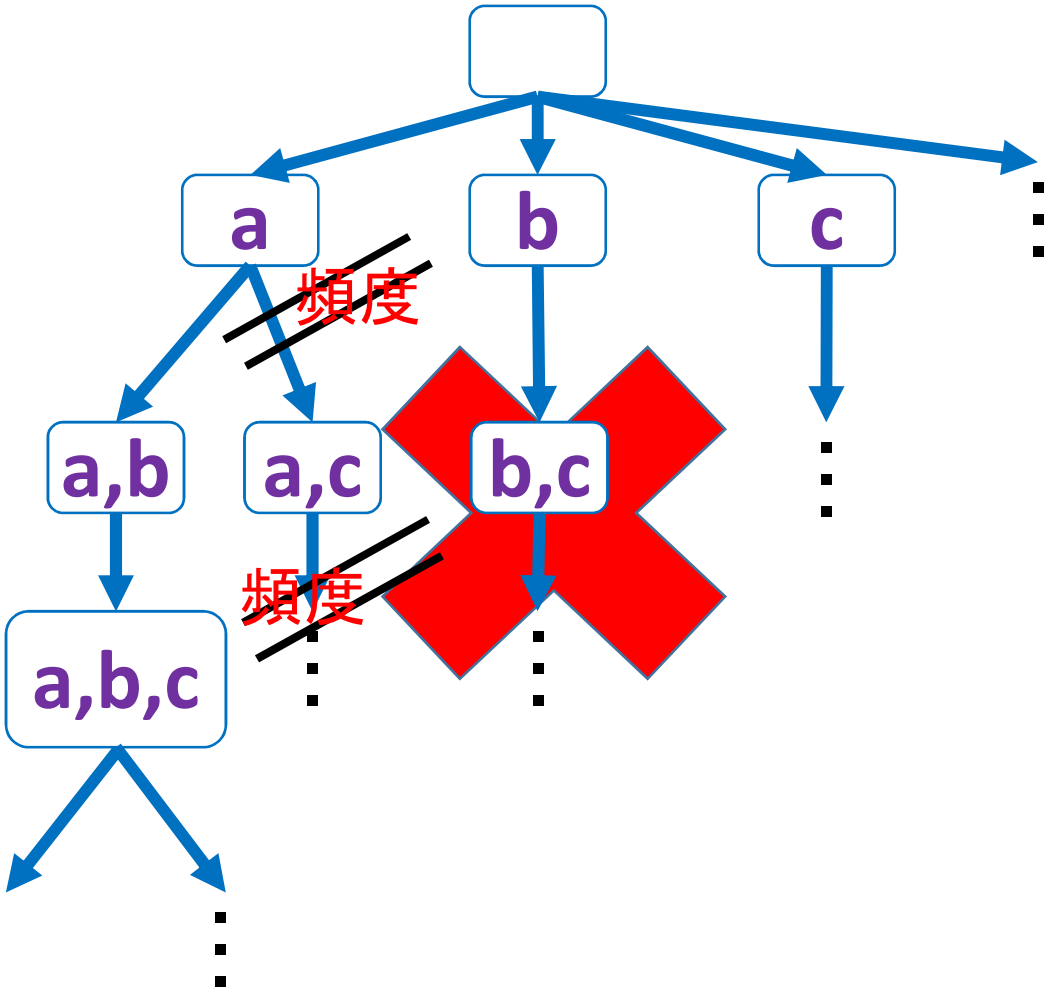
	Tr1	Tr2	Tr3	・・・
b	500	100	400	・・・
c	0	300	600	・・・
d	100	200	0	・・・
e	0	0	500	・・・
⋮	⋮	⋮	⋮	⋮
合計	800	1000	1500	

最小金額 ≥ 上界値



左の枝刈り

	Tr1	Tr2	Tr3	・・・
a	600	400	800	
b	500	100	400	・・・
c	0	300	600	・・・
d	100	200	0	・・・
e	0	0	500	・・・
⋮	⋮	⋮	⋮	⋮
合計				

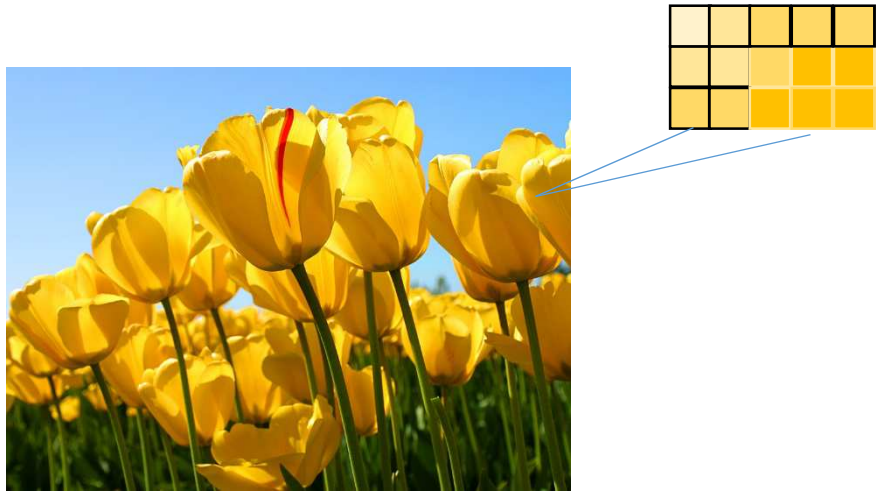


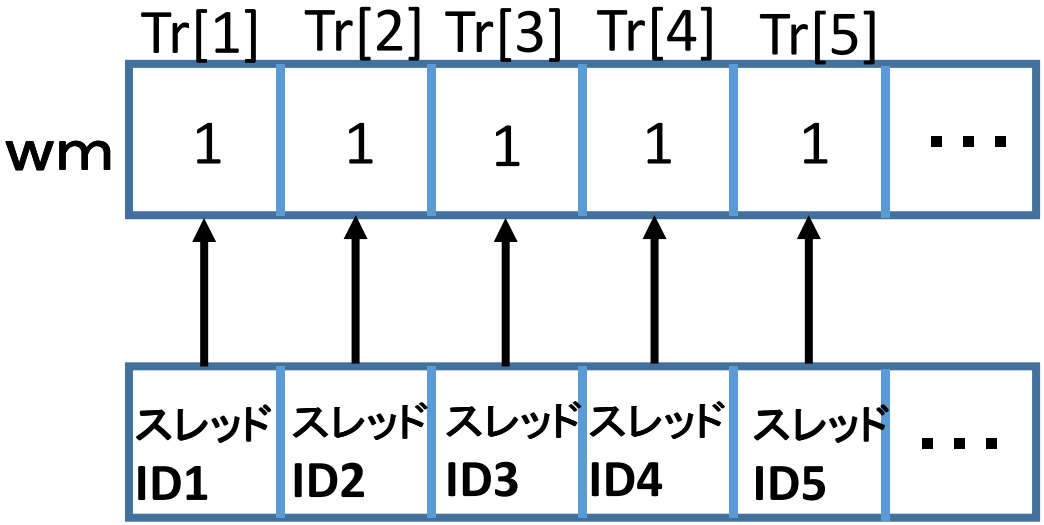
ここでちょっと一息

これから本番です！

GPGPU

GPGPU : General-Purpose computing on
Graphics Processing Units
GPU(Graphics Processing Units)を
汎用目的に使用






```
Tr[ threadIdx.x ] = 1;
```

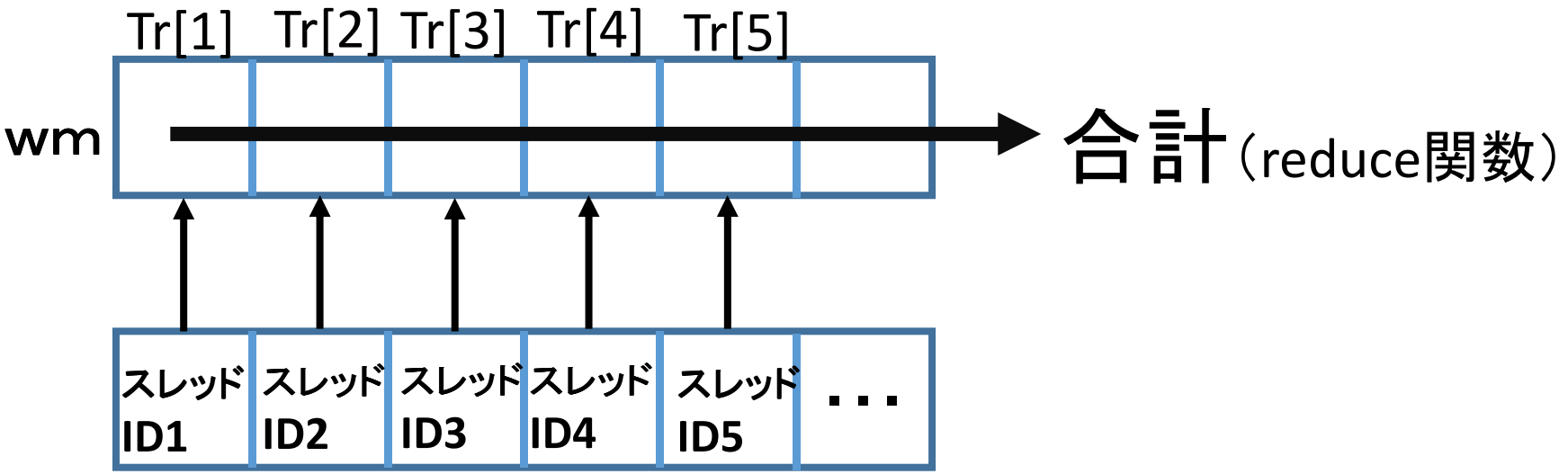
配列の添え字として利用し，
並列化することが多い

実装

CUDA:GPU向けの統合開発環境、C言語をベースに拡張

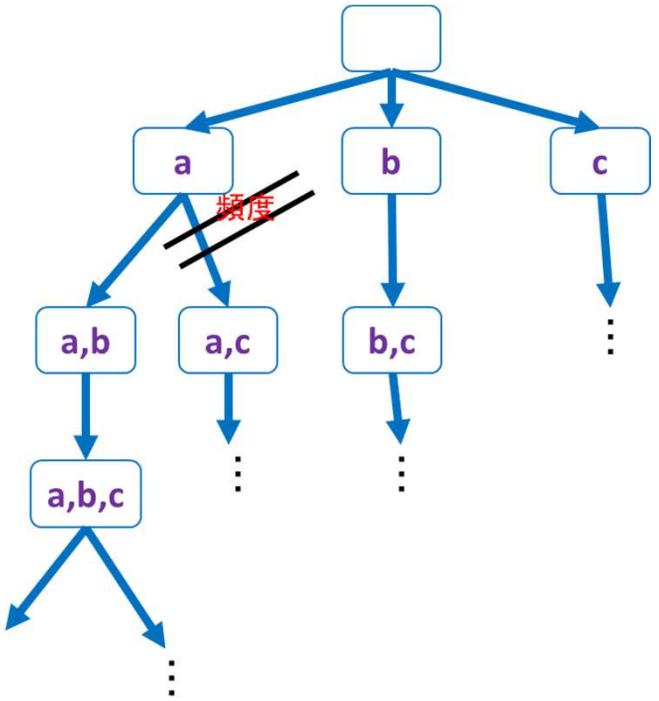
・現在のパターンに対してトランザクションごとに評価値を並列に計算

1. 上界値の計算, パターンの評価値
2. 左右の枝刈りチェック

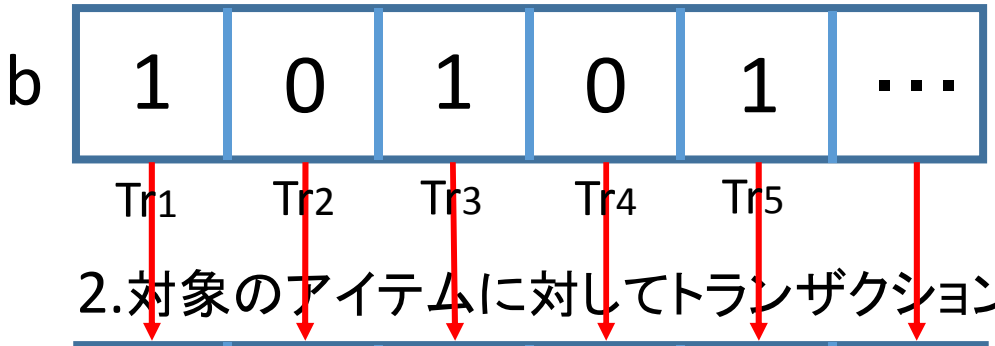


実装

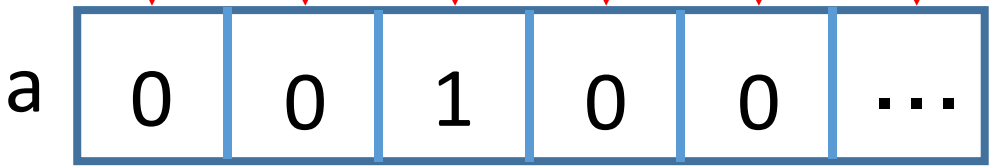
2. 左の枝刈り，右の枝刈りの条件確認



1.現在のパターンに対してトランザクション
ごとに1でビットを立てる



2.対象のアイテムに対してトランザクションごとに0でビットを立てる



0・・・aとb両方持っている
又はaのみ持っている

1・・・bのみ持っている

3.全て0になれば,カバー出来てることが分かるので枝刈りを行う

実験結果

性能

GPU : GPU nVIDIA GeForce GT680 2GB

CPU : MICRO INTEL 2011 Core i7 - 3820 3.6GHz

データセット1 : 中国の小売業のデータが10万件

データセット2 : Twitterのデータ3千万件

ツイッターからネガティブな用語を抽出（評価値：ネガティブ度合い）

※下記の辞書使用

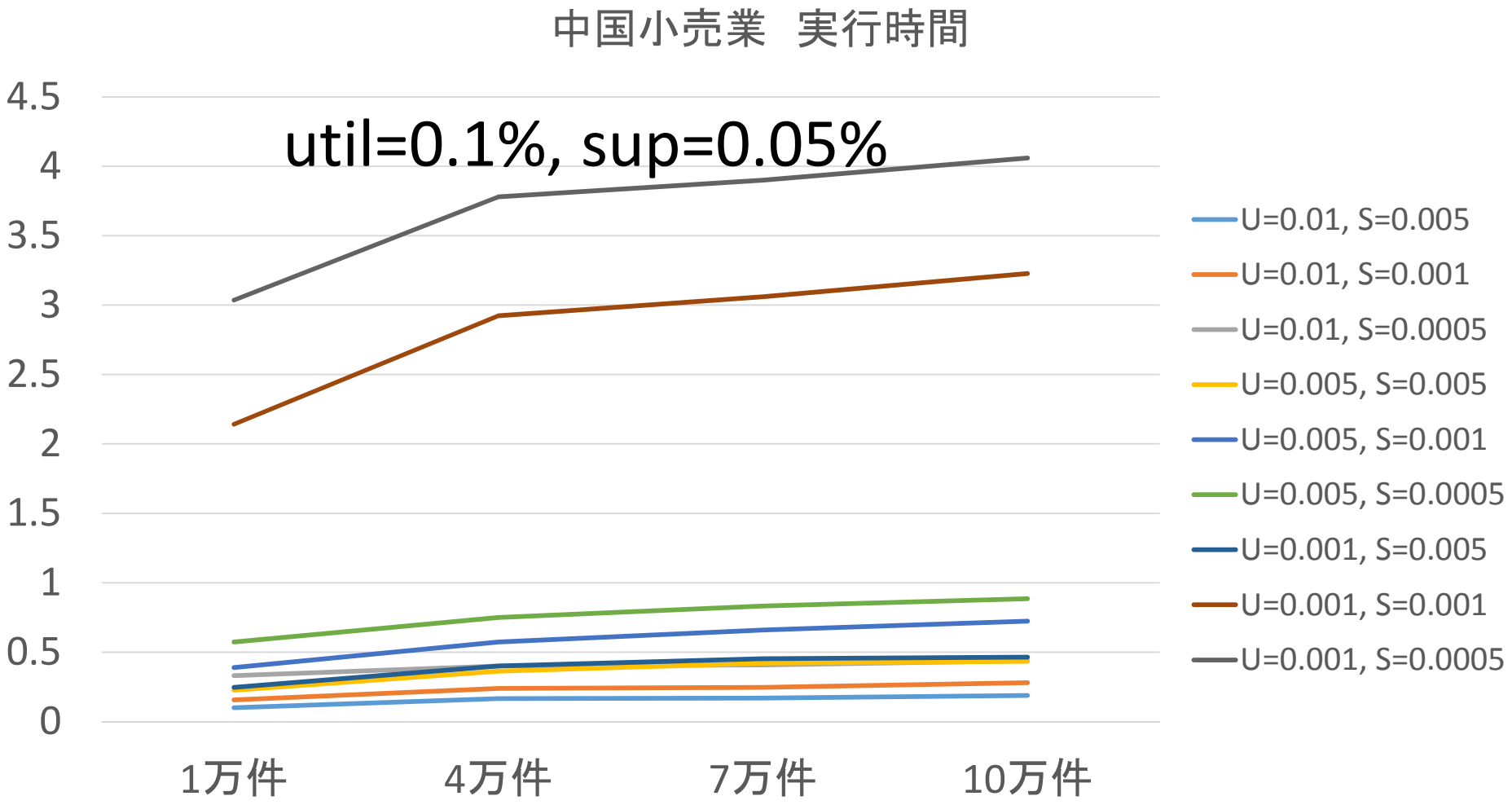
※高村大也, 乾孝司, 奥村学
"スピンモデルによる単語の感情極性抽出
", *情報処理学会論文誌ジャーナル*, Vol.47
No.02 pp. 627--637, 2006.

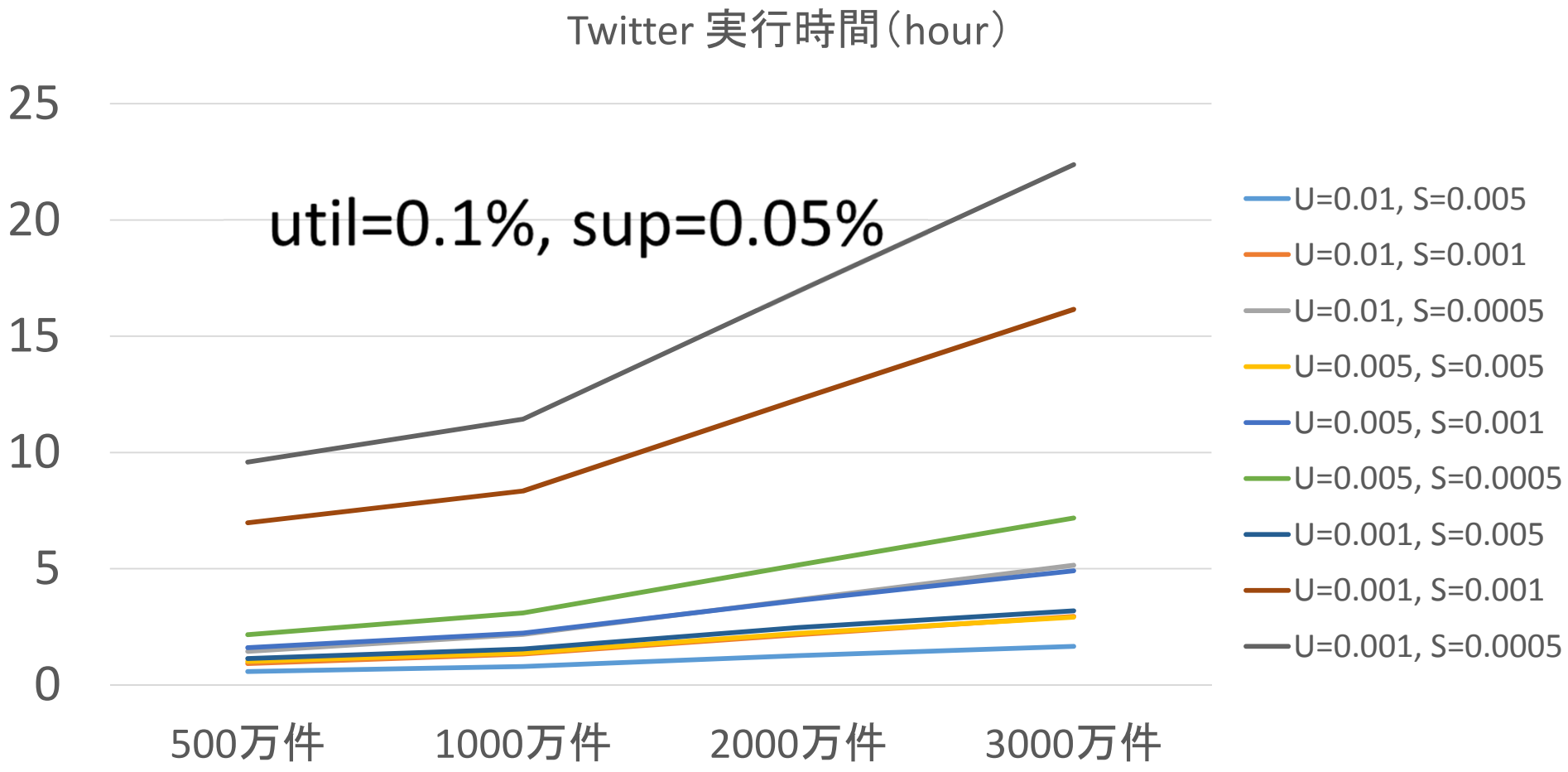
畜生道 : 0.990359

愚か : 0.999303

グロテスク : 0.996421

ネガティブ単語　　点数


中国小売業　実行時間
util=0.1%, sup=0.05%
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
1万件
4万件
7万件
10万件
U=0.01, S=0.005
U=0.01, S=0.001
U=0.01, S=0.0005
U=0.005, S=0.005
U=0.005, S=0.001
U=0.005, S=0.0005
U=0.001, S=0.005
U=0.001, S=0.001
U=0.001, S=0.0005


Twitter 実行時間（hour）
util=0.1%, sup=0.05%
25
20
15
10
5
0
500万件
1000万件
2000万件
3000万件
U=0.01, S=0.005
U=0.01, S=0.001
U=0.01, S=0.0005
U=0.005, S=0.005
U=0.005, S=0.001
U=0.005, S=0.0005
U=0.001, S=0.005
U=0.001, S=0.001
U=0.001, S=0.0005

結論

- 飽和・高価値アイテム集合マイニングのGPGPUによる実装
- 中国の小売業のデータ　10万件
- Twitterデータ3千万件

今後の課題

・探索アルゴリズムの高速化

・大規模な実験を目指す