所属部門	知能情報アーキテクチャ	指導教員	江本 健斗　准教授
学生番号	14231025	氏　名	栗木　駿一
論文題目	大規模グラフ計算用言語 Fregel の 二部グラフマッチングへの応用と拡張		

1　はじめに

近年，SNS におけるユーザー間の関係や，Web ページ間のリンクなどの，大規模なグラフの解析を行う機会が増加してきている．これを受けて，大規模グラフ計算用言語 Fregel が提案された [1]．Fregel は頂点ごとに独立して計算を実行する頂点主体モデルを採用しており，頂点が実行する処理のみを記述することでグラフ計算を並列に行うことができる．しかし，Fregel においての実際のグラフ計算問題の記述例は少ない．

本研究は，Fregel の記述例を増やしてその有用性を検証するために，二部グラフマッチングの Fregel での記述を行った．また，現状の Fregel では機能が不足していたため，Fregel の拡張も同時に行った．

2　二部グラフマッチングの頂点主体計算

二部グラフマッチングとは，頂点が左右の二つの集合に分けられ，それぞれの集合内の頂点間には辺が存在しないようなグラフ（二部グラフ）において，各頂点が高々一つの（他方の）頂点と繋がるような辺の部分集合，ないし，それを求める計算のことである．

これを頂点主体で行うには，まず，左集合の頂点がランダムに右集合の頂点を一つ選択する．次に，右集合の頂点が，左集合の頂点から選ばれたかを調べ，選ばれていればその頂点を選んだ左集合の頂点の中から一つをランダムに選択する．最後に，左集合の頂点が右集合の頂点から選ばれているか確認し，選ばれていればその頂点とマッチングする．これを全頂点がマッチングするか，選択できる頂点が無くなるまで繰り返す．

3　Fregel による記述と拡張の提案

本研究で，実際に二部グラフマッチングを Fregel で記述したものが図 1 である．入力グラフに ss0，ss1，ss2 の 3 つの操作を順に繰り返し行っていくことで，二部グラフマッチングの計算を実現している．

この計算の記述には，リストからランダムに要素を取り出す，といった集約関数が必要となる．しかし，現状の Fregel ではその機能が不足していたため，本研究で新しい集約関数 random を導入するという拡張を行った．これは，random x list と記述することで，list が空ならば x を返し，list に要素があるならばその中からランダムに一つ取り出す，といった関数である．

```
1  bipartitematching g =
2   let init v = (-1);
3       ss0 v = if(vid v > size/2 && val v.^set == -1)
4               then random (-1) [vid u|(e,u)<-is v,
5                                 val u.^set == -1]
6               else val v.^set);
7       ss1 v = if(vid v <= size/2 && val v.^set == -1)
8               then random (-1) [vid u|(e,u)<-is v,
9                                 val u.^set == vid v]
10              else val v.^set);
11      ss2 v = if(vid v > size/2 && val v.^set /= -1)
12              then random (-1) [vid u|(e,u)<-is v,
13                                val u.^set == vid v]
14              else val v.^set);
15      step g = let g1 = gmap ss0 g;
16                   g2 = gmap ss1 g1;
17                   g3 = gmap ss2 g2
18               in g3
19    in giter init step Fix g
```

図 1 :二部グラフマッチングの Fregel コード

4　実験と評価

図 1 のコードに対し，その並列実行を確認する実験を行った．入力にはランダム生成した，各頂点が 10 本の辺を持つ二部グラフを使用した．頂点数毎に，計算機の台数に応じた速度向上の様子を図 2 に示す．

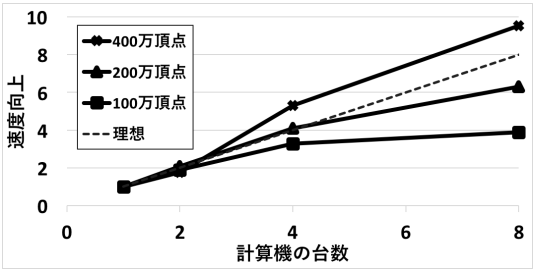

図 2 :並列実行による速度向上の様子

図 2 から，計算機の台数が増加するに連れて速度が向上していることがわかる．また，頂点数が増加するに連れて速度向上が改善しているため，より大規模なグラフほど並列化の効果が顕著になることが分かる．よって，本研究で拡張した機能を用いたグラフ計算においても，十分な並列性能が得られることが分かった．さらに，少しの機能拡張により二部グラフマッチングを記述できたことで，Fregel の記述性も確認された．

参考文献

[1] Kento Emoto, Kiminori Matsuzaki, Zhenjiang Hu, Akimasa Morihata, and Hideya Iwasaki. Think like a vertex, behave like a function! a functional DSL for vertex-centric big graph processing. In *Proceedings of ICFP 2016*, pages 200–213, 2016.