Parallelizing Sequential Graph Computations

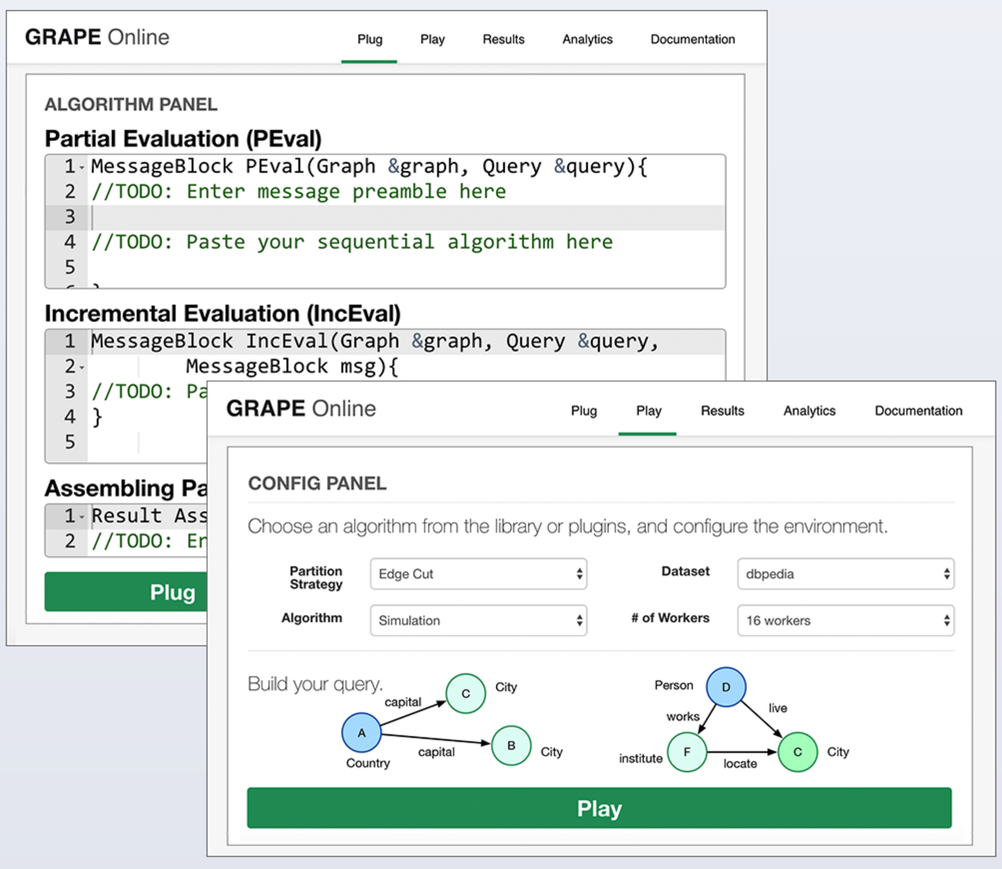
Wenfei Fan[1,2], Jingbo Xu[1,2], Yinghui Wu[3], Wenyuan Yu[2],
Jiaxin Jiang[4], Zeyu Zheng[5], Bohan Zhang[5], Yang Cao[1], Chao Tian[1,2]

[1]Univ. of Edinburgh [2]Beihang Univ. [3]Washington State Univ. [4]Hong Kong Baptist Univ. [5]Peking Univ.

INTRODUCTION

GRAPE, parallelizing existing sequential algorithms as a whole, and guaranteeing convergence and correctness when the sequential algorithms provided are correct.

- Sequential graph algorithms can be "plugged into" GRAPE with minor additions, and get parallelized.
- Foundation: a simultaneous fixed point computation with **partial evaluation** and **incremental evaluation**.



MOTIVATION

- It is nontrivial for one to learn how to program in the new parallel models, e.g., "think like a vertex".
- Graph computations have been studied for decades, and a number of sequential graph algorithms are already in place. Can we use them without recasting?
- Do existing parallel graph engines guarantee termination and correctness?

GRAPE API

Data partitioned parallelism (shared-nothing architecture).

Fragmented graph $G = (G_1, \ldots, G_n)$, distributed to workers.

GRAPE API: three core functions for a graph query class Q

PEval: a (existing) sequential algorithm for Q, for partial evaluation;

IncEval: a (existing) sequential incremental algorithm for Q;

Assemble: a sequential algorithm (taking a union of partial results).

FOUNDATION OF GRAPE

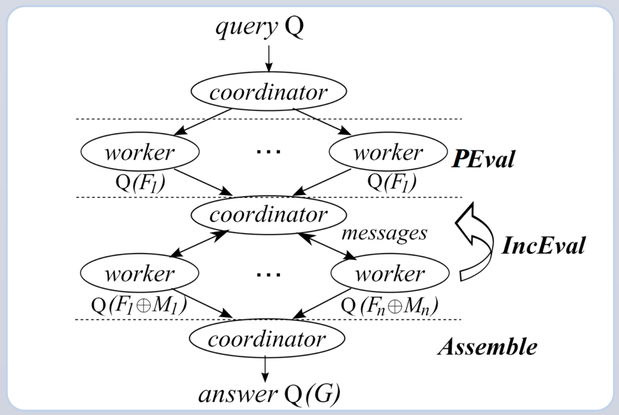
A fixpoint computation

$\Phi(R_1, \ldots, R_n)$

$R_i^0 = \text{PEval}(Q, G_i)$

$R_i^{r+1} = \text{IncEval}(Q, R_i^r, G_i, M_i)$

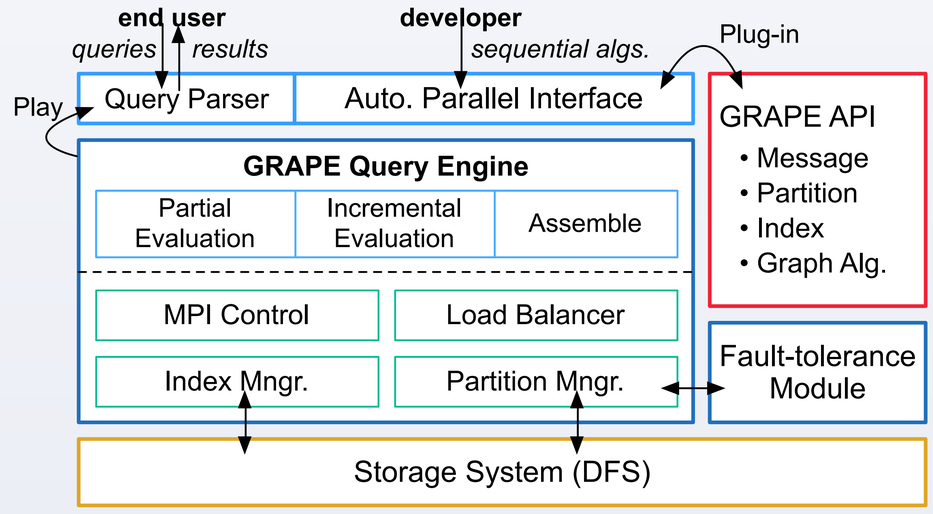
- R_i^r : partial results at processor i in round r
- M_i : message to processor i



Assurance Theorem: termination and correctness guaranteed if the sequential algorithms provided are correct, and M_i is "monotonic"!

Simulation Theorem: MapReduce, BSP (bulk synchronous parallel) and PRAM models are optimally simulated by GRAPE

IMPLEMENTATION AND FEATURES

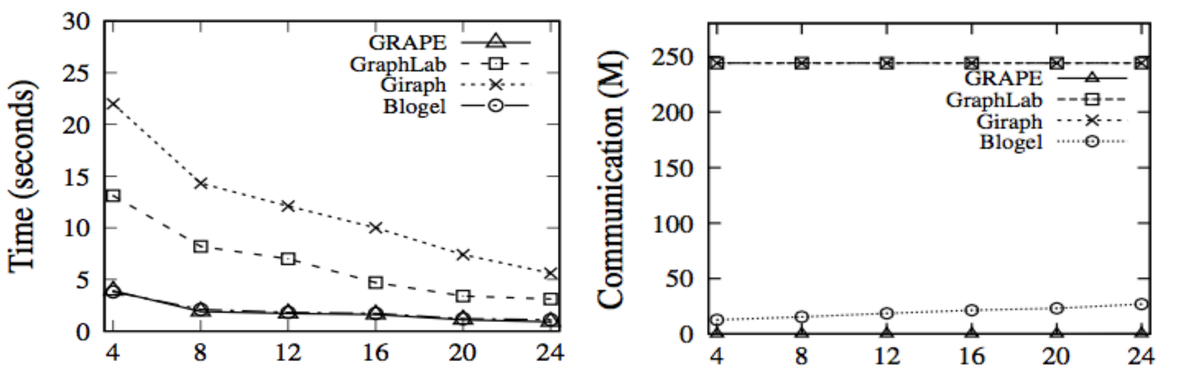


- **Ease of programming.** Only need to provide three (existing) sequential (incremental) algorithms for Q with minor additions.
- **Semi-automated parallelization.** Guarantee to converge at correct answers under a monotonic condition, if the three sequential algorithms provided are correct.
- **Graph-level optimization.** GRAPE inherits all optimization strategies available for sequential algorithms and graphs.
- **Scale-up.** GRAPE achieves comparable performance to the state-of- the-art graph systems.

PERFORMANCE

Sequential Algorithms and their applications: (1) **SSSP** (traffic analysis); (2) **Connected Component** (social analysis); (3) **Graph Simulation** (social recommendation); (4) **Keyword Search/ Subgraph Isomorphism**(Web/knowledge mining); (5) **Collaborate Filtering** (recommendation, machine learning).

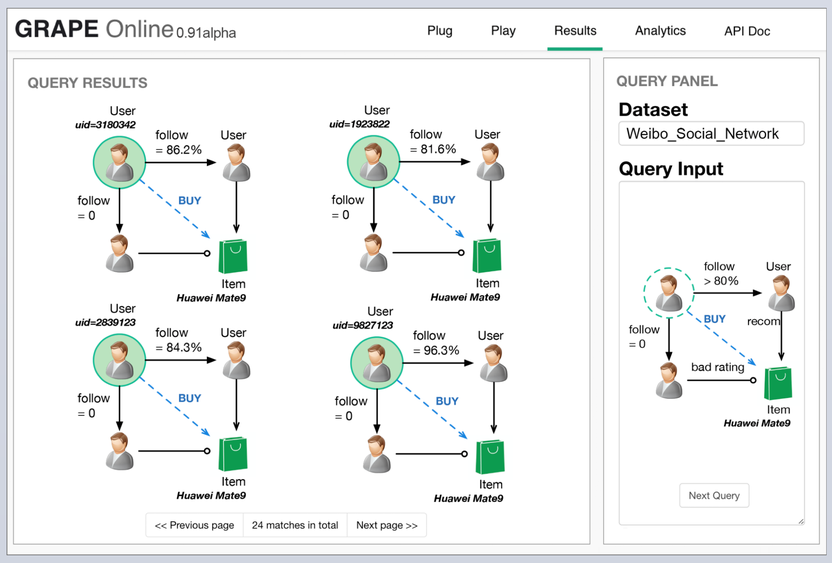
2 times faster than Giraph, orders of magnitudes less data shipment



(c) Varying n: SSSP (DBpedia) (c) Varying n: SSSP (DBpedia)

APPLICATION SCENARIO

Social media marketing with Graph Pattern Association Rules.



Graph Pattern Association Rules (GPARs): $Q(x,y) \Rightarrow p(x,y)$,

Left figure shows a GPAR ϕ: if among the people followed by x_o (a) at least 80% of them recommend Huawei Mate 9, and (b) no one gives it a bad rating, then the chances are that x_o will buy this mobile phone.

CONCLUSION

GRAPE: make parallel graph computations accessible to users who know conventional graph algorithms from undergraduate textbooks.

Welcome to play with GRAPE (alpha): http://grapedb.io