Отчет по практическому заданию по курсу «Параллельная обработка больших графов» Параллельная реализация алгоритма дельта-степпинг решения задачи SSSP

Подготовил: Куприй Роман Михайлович, 523 группа
Вариант реализации: MPI-1, Bsend

1. Вычислительные ресурсы

Для выполнения задания использовался вычислительный комплекс IBM Polus, состоящий из 5 узлов (1 фронтенд, 1 недоступный и 3 доступных вычислительных узла). На каждом вычислительном узле по 2 процессора с 10 ядрами и общей ОЗУ 256Гб.

Решение выполнено с использованием C++ и технологии распараллеливания MPI, с версией компилятора gcc 9.3.1 (доступен на полюсе через devtoolset-9) и использованием планировщика LSF для запуска задач на нескольких узлах.

2. Решение

Алгоритм реализован с использованием функции MPI_Bsend для обмена сообщениями между процессами. Размер буфера удваивается при необходимости, также используется свой тип данных для переслыки: структура message, содержащая глобальный номер вершины и вес исходящего ребра. Также задействованы по одной глобальной коммуникации MPI_Alltoall для определения размера входящих и исходящих сообщений, а также MPI_Allreduce для обмена информацией об обработке текущей корзины (bucket-a) между процессами.

Ключевыми вопросами для достижения требуемой производительноси стало определение параметра delta (при неудачном выборе этого параметра производительность снижается в разы) и перемещение обработанных вершин (точнее расстояния до неё) между корзинами. Эта операция подразумевает удаление вершины из старой корзины при каждой релаксации, а значит требует эффективной реализации. Использование операций с линейной сложностью приводит к снижению производительности на порядки. В настоящей реализации используется контейнер set который позволяет удалять вершину за амортизированную константу. Потенциально можно было бы реализовать эту процедуру за константу в обмен на использование дополнительной памяти.

3. Корректность решения

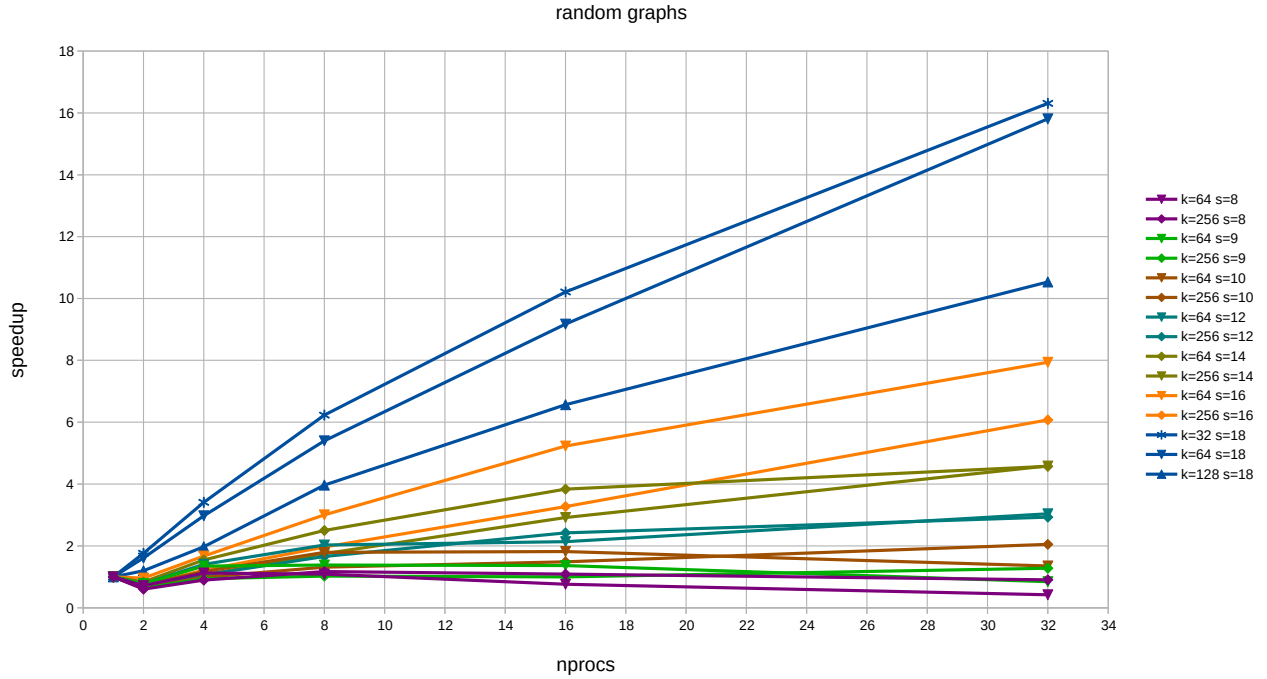
Пример пары сравнений: для random и rmat графа

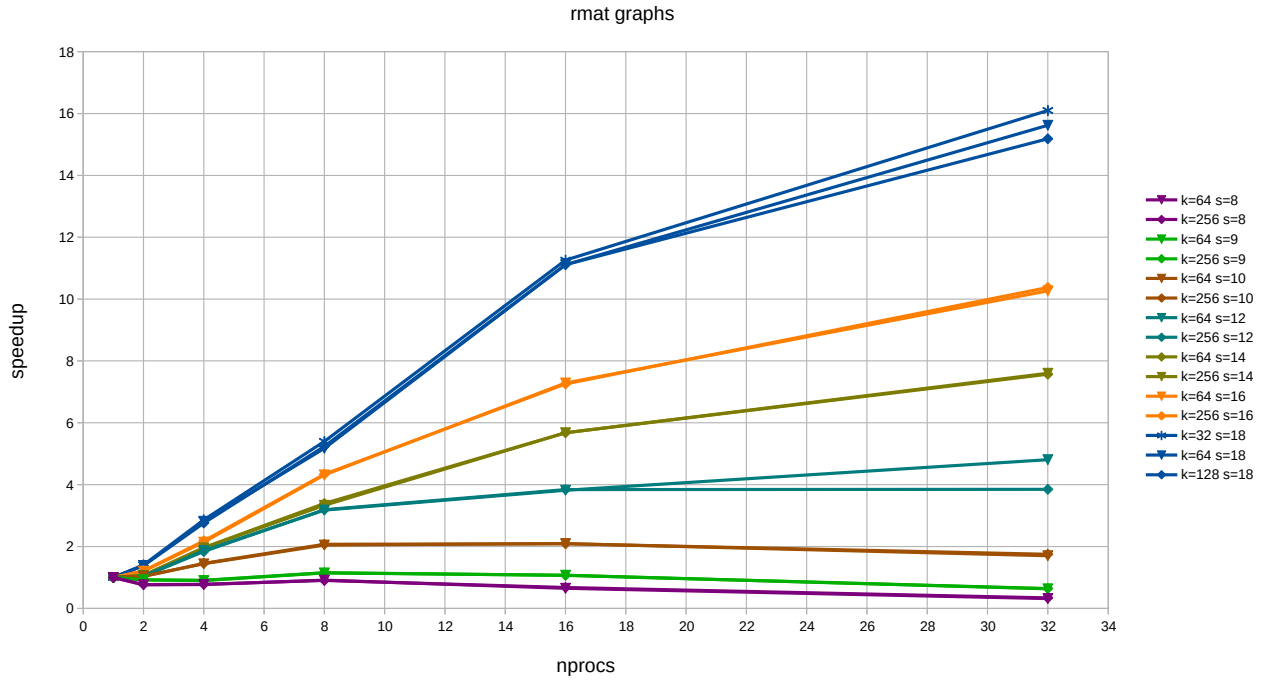
```
[edu-cmc-sqi21-06@polus-ib sem10]$ mpirun -np 8 ./gen_random_mpi -nRoots 10 -s 10 -k 128
[edu-cmc-sqi21-06@polus-ib sem10]$ ./sssp_serial -in random-10 -nFiles 8 -nIters 10
start algorithm iterations...
0: numTraversedEdges = 262144, time = 0.001103 s, perf = 237.6092 MTEPS
1: numTraversedEdges = 262144, time = 0.001075 s, perf = 243.8413 MTEPS
2: numTraversedEdges = 262144, time = 0.001070 s, perf = 244.9614 MTEPS
3: numTraversedEdges = 262144, time = 0.001062 s, perf = 246.8102 MTEPS
4: numTraversedEdges = 262144, time = 0.001070 s, perf = 245.0436 MTEPS
5: numTraversedEdges = 262144, time = 0.001066 s, perf = 245.9335 MTEPS
6: numTraversedEdges = 262144, time = 0.001067 s, perf = 245.5709 MTEPS
7: numTraversedEdges = 262144, time = 0.001062 s, perf = 246.7635 MTEPS
8: numTraversedEdges = 262144, time = 0.001070 s, perf = 245.0736 MTEPS
9: numTraversedEdges = 262144, time = 0.001066 s, perf = 246.0231 MTEPS
algorithm iterations finished.
random-10: nIters = 10 SSSP performance min = 237.6092 avg = 244.7630 max = 246.8102 MTEPS
[edu-cmc-sqi21-06@polus-ib sem10]$ mv random-10.v random-10.v.ref
[edu-cmc-sqi21-06@polus-ib sem10]$ mpirun -np 8 ./sssp_mpi -in random-10
Start sssp with delta=0.0390625
0: numTraversedEdges = 262144, time = 0.007542 s, perf = 34.7584 MTEPS
Start sssp with delta=0.0390625
1: numTraversedEdges = 262144, time = 0.006538 s, perf = 40.0960 MTEPS
Start sssp with delta=0.0390625
2: numTraversedEdges = 262144, time = 0.004065 s, perf = 64.4875 MTEPS
Start sssp with delta=0.0390625
3: numTraversedEdges = 262144, time = 0.004175 s, perf = 62.7898 MTEPS
Start sssp with delta=0.0390625
4: numTraversedEdges = 262144, time = 0.004308 s, perf = 60.8507 MTEPS
Start sssp with delta=0.0390625
5: numTraversedEdges = 262144, time = 0.005612 s, perf = 46.7122 MTEPS
Start sssp with delta=0.0390625
6: numTraversedEdges = 262144, time = 0.006921 s, perf = 37.8776 MTEPS
Start sssp with delta=0.0390625
7: numTraversedEdges = 262144, time = 0.004633 s, perf = 56.5825 MTEPS
Start sssp with delta=0.0390625
8: numTraversedEdges = 262144, time = 0.006576 s, perf = 39.8634 MTEPS
Start sssp with delta=0.0390625
9: numTraversedEdges = 262144, time = 0.006523 s, perf = 40.1868 MTEPS
random-10.0: nIters = 10 SSSP performance min = 34.7584 avg = 48.4205 max = 64.4875 MTEPS
[edu-cmc-sqi21-06@polus-ib sem10]$ ./compare random-10.v.ref random-10.v
norm = 0
Ok
[edu-cmc-sqi21-06@polus-ib sem10]$
```

```
[edu-cmc-sqi21-06@polus-ib sem10]$ mpirun -np 8 ./gen_RMAT_mpi -s 8 -k 128 -nRoots 10
[edu-cmc-sqi21-06@polus-ib sem10]$ ./sssp_serial -in rmat-8 -nFiles 8 -nIters 10
start algorithm iterations...
0: numTraversedEdges = 8192, time = 0.000080 s, perf = 102.7983 MTEPS
1: numTraversedEdges = 8192, time = 0.000077 s, perf = 106.1745 MTEPS
2: numTraversedEdges = 8192, time = 0.000077 s, perf = 106.7070 MTEPS
3: numTraversedEdges = 8192, time = 0.000077 s, perf = 105.8207 MTEPS
4: numTraversedEdges = 8192, time = 0.000076 s, perf = 107.1606 MTEPS
5: numTraversedEdges = 8192, time = 0.000075 s, perf = 109.0856 MTEPS
6: numTraversedEdges = 8192, time = 0.000075 s, perf = 109.2631 MTEPS
7: numTraversedEdges = 8192, time = 0.000073 s, perf = 111.7797 MTEPS
8: numTraversedEdges = 8192, time = 0.000076 s, perf = 107.9898 MTEPS
9: numTraversedEdges = 8192, time = 0.000076 s, perf = 108.4976 MTEPS
algorithm iterations finished.
rmat-8: nIters = 10 SSSP performance min = 102.7983 avg = 107.5277 max = 111.7797 MTEPS
[edu-cmc-sqi21-06@polus-ib sem10]$ mv rmat-8.v rmat-8.v.ref
[edu-cmc-sqi21-06@polus-ib sem10]$ mpirun -np 8 ./sssp_mpi -in rmat-8
Start sssp with delta=0.3125
0: numTraversedEdges = 8192, time = 0.000835 s, perf = 9.8115 MTEPS
Start sssp with delta=0.3125
1: numTraversedEdges = 8192, time = 0.000595 s, perf = 13.7659 MTEPS
Start sssp with delta=0.3125
2: numTraversedEdges = 8192, time = 0.000595 s, perf = 13.7659 MTEPS
Start sssp with delta=0.3125
3: numTraversedEdges = 8192, time = 0.000683 s, perf = 11.9929 MTEPS
Start sssp with delta=0.3125
4: numTraversedEdges = 8192, time = 0.000708 s, perf = 11.5689 MTEPS
Start sssp with delta=0.3125
5: numTraversedEdges = 8192, time = 0.000683 s, perf = 11.9971 MTEPS
Start sssp with delta=0.3125
6: numTraversedEdges = 8192, time = 0.000782 s, perf = 10.4755 MTEPS
Start sssp with delta=0.3125
7: numTraversedEdges = 8192, time = 0.000615 s, perf = 13.3229 MTEPS
Start sssp with delta=0.3125
8: numTraversedEdges = 8192, time = 0.000640 s, perf = 12.7969 MTEPS
Start sssp with delta=0.3125
9: numTraversedEdges = 8192, time = 0.000614 s, perf = 13.3436 MTEPS
rmat-8.0: nIters = 10 SSSP performance min = 9.8115 avg = 12.2841 max = 13.7659 MTEPS
[edu-cmc-sqi21-06@polus-ib sem10]$ ./compare rmat-8.v.ref rmat-8.v
norm = 0
Ok
[edu-cmc-sqi21-06@polus-ib sem10]$
```

4. Результаты

Для маленьких графов (меньше 2^{10} вершин) реализованный алгоритм не показывает свойство сильной масштабируемости, однако, при увеличении числа вершин растёт эффективность, позволяя получить всё большее ускорение в сравнении с 1 процессом.





Это справедливо и для случайных графов и для RMAT графов. Такое поведение проиллюстрировано на графиках выше. Тестовые запуски проводились для

графов размером от 2^8 до 2^{18} вершин со средней глубиной вершин 64 и 256 (32, 64 и 128 для последнего графа). Графы одного размера отмечены одним цветом.

Стоит отметить, что для RMAT сгенерированных графов тот же выбор параметра delta приводит к значительно меньшей производительности. Также, в отличие от случайно сгенерированных графов, где на производительность параллельного алгоритма сильно влияет средняя глубина вершины, для RMAT графов такого влияния практически полностью нет.

В итоге, параллельный алгоритм удовлетворяет свойству сильной масштабируемости для достаточно больших графов (вместе с этим растёт и ускорение относительно последовательной версии). Реализация масштабируема по памяти. В качестве возможных оптимизаций могут быть: луший подбор параметра delta, оптимизация хранения и перемещения вершин в bucket-ах .