Course: Comtemporary Algorithms T.II/2019-20

Lecture 10: Parallel Algorithms IV

5 February 2020

Lecturer: Dr. Kanat Tangwongsan *Scribe: Suchanun P.& Suchanuch P.*

1 Prime Sieves

1.1 Finding primes $\leq$ n

$is\_prime(x)$ returns whether or not x is prime.
Sequential: $W = O(\sqrt{x})$
Parallel: $W = O(\sqrt{x}), S = O(\log x)$

$find\_primes$ returns all primes up to n.

Algorithm 1: find_primes(n)

for *i in range(2, n+1)* **do**
  flags[i] = is_prime(i);
end
return flags.filter(lambda x: x) ; // where flags store $True$

$W = O(n\sqrt{n})$ and $S = O(\log n)$

1.2 Sieve of Eratosthenes

To find all primes up to n. Generate a list of integers from 2 to n. Say $n = 30$

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

The first number in the list is 2. Cross out all multiple of 2 from the list.

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~ 19 ~~20~~ 21 ~~22~~ 23 ~~24~~ 25 ~~26~~ 27 ~~28~~ 29 ~~30~~

The next number in the list is 3. Cross out all its multiple from the list.

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~ ~~21~~ ~~22~~ 23 ~~24~~ 25 ~~26~~ ~~27~~ ~~28~~ 29 ~~30~~

The next number not yet crossed out in the list after 3 is 5.
Repeat the same process until we cross the multiples of $\sqrt{n}$

2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~ ~~21~~ ~~22~~ 23 ~~24~~ ~~25~~ ~~26~~ ~~27~~ ~~28~~ 29 ~~30~~

The numbers not crossed out at this point are all primes.

```
2  3     5     7           11    13          17    19          23                   29
```

$T = \sum_{p=primes \le n} \frac{n}{p} \le n(\log\log n + const) = O(n \log\log n)$

Algorithm 2: find_primes(n)

if *n<2* **then**
  return []
end
sqrtn = sqrt(n);
low_primes = find_primes(sqrtn) ; // $W : O(\sqrt{n}),\ S : O(\sqrt{n})$
flags = [True]*n;
pfor *p in lowprimes* **do** // $W : O(n \log\log n),\ S : O(1)$
  pfor *(i = sqrtn/p; i< n/p; i++)* **do**
    flags[$p * i$] = False;
  end
end
high_primes = filter(range(sqrt+1, n+1), lambda x: flags[x]) ; // $W : O(n),\ S : O(\log n)$
return low_primes + high_primes

Work and Span

$W(n) = W(\sqrt{n}) + O(n \log\log n) = O(n \log\log n)$
$S(n) = S(\sqrt{n}) + O(\log n) = \log n + \frac{1}{2}\log n + \frac{1}{4}\log n + \ldots = O(\log n)$

2 MST

Given (V, E, w), get MST of minimum weight.
Boruvka (1926) - based on Light Edge Rule
Theorem: Let $G = (V, E, w)$ be a connected, undirected graph qith distinct edge weights. For any nonempty $U \subset V$, the minimum weight edge between U and $V \setminus U$ is in the MST of G.
Observation: The min edge of each vertex appears in the MST.
Claim1: The min edge form a forest (with no cycles)
Claim2: # nodes contracted $\ge \frac{n}{2}$

Algorithm 3: MST($G = (V, E)$)

if $|V| == 1$ **then**
  return;
end
every vertex picks its min edge $\rightarrow$ mindEdges; add this to final MST;
 // $W : O(n)\ S : O(1)$
run tree-contract on minEdges $\rightarrow G' = (V', E')$, ; // $W : O(m),\ S : O(\log^2 n)$
MST(G') ; // $W :\le W(\frac{n}{2}, m),\ S : S(\frac{n}{2}, m)$

Work and Span
$W(n,m) \leq W(\frac{n}{2}, m) + O(n) + O(m) \leq O(m \log n + n)$
$S(n,m) = S(\frac{n}{2}, m) + O(\log^2 n) = O(\log^3 n)$

3 Connectivity

Given $G = (V, E)$, want to assign labels $l : v \to \{0, ...\}$,
such that $l(u) = l(v) \to u$ is connected to v.

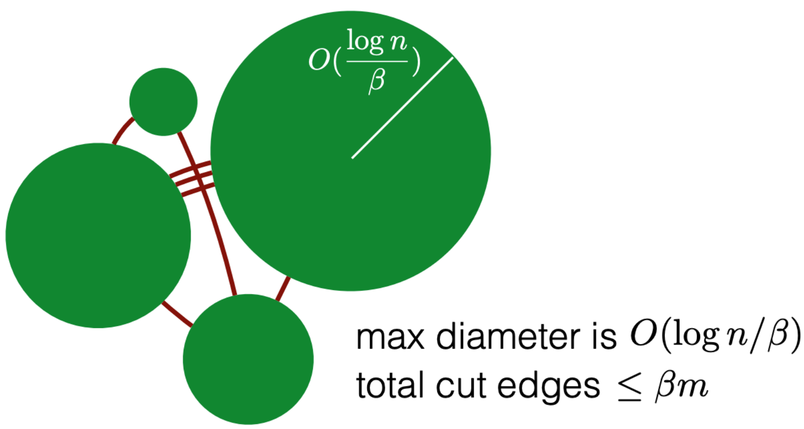
Sequencial BFS/DFS can do this in $O(m + n)$.

3.1 Low-diameter decomposition

Goal: decompose V into a set of clusters s.t.
1. the number of inter-cluster edges is "small"
2. diameter of each cluster is "small" (log(n))

Def: a $(\beta, d)-$decomposition, $0 < \beta < 1$, is a partition of V into $V_1, V_2, ..., V_k$ such that

- total number of edges across components $\leq \beta m$ (few inter-component edges)
- the shortest path between any 2 vertices in $u, v \in V_i$, using only vertices in V_i is at most d. (strong diameter)



Theorem: Parallel low-diameter decomposition can find $(\beta, d)-$ decomposition
where $\beta \leq 1/2$ and $d \in O(\log n/\beta)$ in $O(m)$ work and $O(\log^2 n)$ span with high probability.