

# شبیه سازی صف

ابراهیم بیاگوی

# بهینه سازی ها

- انتخاب زبانی بهتر برای برنامه های بهینه
- استفاده نکردن از اختصاص حافظه پویا و مصرف کمتر حافظه

- استفاده از الگوریتم بهینه تر برای تولید اعداد شبه تصادفی
- بهینه سازی الگوریتم انتخاب (اسلاید بعد)

```
inline int fastrand() {  
    g_seed = (214013 * g_seed + 2531011);  
    return (g_seed >> 16) & 0x7FFF;  
}
```

- استفاده از Release به جای Debug
- نتیجه برای ۱۰ میلیون:

std::rand به نسبت سرعت

- از ۱۰ ثانیه کد اصلی، به ۰.۳ ثانیه

- مصرف گیگابایتی حافظه، به مصرف ۷۰۰ کیلوبایتی

- نگه داشتن تست ها جهت اطمینان از صحت

1.8446744e+19

- دو پیاده سازی (هر دو انجام شد):

محدودیت در uint64\_t

- پیاده سازی بی نهایت با امکان ذخیره حالت

- پیاده سازی با محدودیت uint64\_t ولی با سرعت بیشتر

# انتخاب از بین موارد

## • بهبود الگوریتم انتخاب مورد

بعد

```
class ItemPicker {
    uint8_t* items;
    int* aggregatedPossibilities;
    unsigned long n;

public:

    ItemPicker(std::map<uint8_t, float> possibilities) {
        n = possibilities.size();
        items = new uint8_t[n];
        aggregatedPossibilities = new int[n];
        float aggregation = 0;
        int i = 0;
        for (auto const item : possibilities) {
            aggregation += item.second;
            aggregatedPossibilities[i] = aggregation * RAND_MAX;
            items[i] = item.first;
            i++;
        }
    }

    inline uint8_t pick(int value) {
        for (int i = 0; i < n; ++i)
            if (value <= aggregatedPossibilities[i])
                return items[i];
    }
}
```

قابل بهینه سازی  
در صورت تعداد  
بالای گزینه ها  
برای انتخاب  
(bsearch)

بهینه سازی: ورودی احتمال به صورت عدد صحیح

قبل

```
function* itemPicker(stream, items) {
    for (let value of stream)
        for (let item in items) {
            if (Math.round(value * 1000) / 1000 === 0) {
                yield +item;
                break;
            }
            value -= items[item];
            if (Math.round(value * 1000) / 1000 <= 0) {
                yield +item;
                break;
            }
        }
}
```

# اجرای ۱۰ میلیارد مورد در ۳۳۰ ثانیه

```
C:\Users\Ebrahim\Desktop\SimulationOptimized\build\Release\main.exe
Press space to see the simulation state...

Result of the simulation with 8180290400 cases:
Elapsed time: 270.398000s
Average waiting time: 1.178842
Ratio of waited customers: 0.354659
Ratio of no customer times: 0.288887
Service duration average: 2.617707
Average of customers entering leaps: 3.681142
Average of waited customers waiting times: 3.323873
Average of customers being in system time: 3.796553

Result of the simulation with 10000000000 cases:
Elapsed time: 330.365000s
Average waiting time: 1.441078
Ratio of waited customers: 0.433553
Ratio of no customer times: 0.288887
Service duration average: 3.200013
Average of customers entering leaps: 4.500009
Average of waited customers waiting times: 3.323881
Average of customers being in system time: 4.641091

PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED! PASSED!
```

# انجام در متلب

استفاده از امکانات متلب و تاکید بر تولید و استفاده از آرایه/ماتریس‌ها

```
simulation.m x +
21- n = 100000000;
22-
23- arrivals = randsample(arrivalDistribution(:, 1)', n, true, arrivalDistribution(:, 2)');
24- services = randsample(serviceDistribution(:, 1)', n, true, serviceDistribution(:, 2)');
25-
26- fprintf('N: %d\n', n);
27- %%%%%%%%%%%%%% QUEUE SIMULATION %%%%%%%%%%%%%%
28- waitingTimeAvg = main(arrivals, services, n);
29- %%%%%%%%%%%%%%
30-
31- arrivalMean = mean(arrivals);
32- arrivalVar = var(arrivals) ^ .5;
33- servicesMean = mean(services);
34- servicesVar = var(services) ^ .5;
35-
36- lambda = 1 / arrivalMean;
37- tau = servicesMean;
38- ro = lambda / (1 / tau);
39- ca = arrivalVar / arrivalMean;
40- cs = servicesVar / servicesMean;
41-
42- fprintf('Expected: %f, Actual: %f\n', ...
43-     ... % Kingman's formula (estimation)
44-     (ro / (1 - ro)) * (((ca ^ 2) + (cs ^ 2)) / 2) * tau, ...
45-     waitingTimeAvg);
```

$$\mathbb{E}(W_q) \approx \left( \frac{\rho}{1 - \rho} \right) \left( \frac{c_a^2 + c_s^2}{2} \right) \tau$$

Command Window

New to MATLAB? See resources for [Getting Started](#).

N: 100000000

fx Expected: 1.659791, Actual: 1.441394