PRODUCTION SCHEDULING PROJECT

Hospital Emergency Room Scheduling

Xiaoyu Guo xg2164
Adrian Luk acl2150

Project Description … never been studied in class

- Simulated from limited data (4 Sundays) gathered at St. Luke's Hospital ER
  - ↳ Determined the distributions for each process to represent the situation.
    - ↳ Defined scheduling problem and proposed solutions
      - ↳ Conducted computational analysis and analyzed results in MATLAB

- Processes in sequence:
  - Triage – Patients see a nurse first
  - Treatment – After receiving treatment by a doctor, patients leave the ER
    - Nurse/Doctor's processing times depend on the criticality of each patient
- Facilities:
  - 2 treatment rooms in the ER
  - 1 nurse
  - 2 doctors
- Patients (jobs):
  - Arrive according to Poisson distribution / Exponential arrival rate
  - 5 criticalities: (5-Critical; 4-High; 3-Medium; 2-Low; 1-Clear/non-critical)
  - Death metrics: determined by release time and a uniformly distributed random variable with expectation that's associated with criticality

Model Assumptions

- Patients arrive according to exponential distribution with lambda = 0.2
  - tA = -log(rand)/lambda
- Each type of patient has a determined chance to arrive
  - 10% cleared patient; 20% low criticality

  20% medium criticality; 30% high criticality; 20% critical
- Processing times:
  - Nurse processing time is random with expected duration of 10 minutes for all patients
  - Doctor processing time depending on criticality of patients
    - Uniformly distributed within doctor processing time bracket, i.e. critical patients has processing time between 75 -95 minutes.
- Death Metrics:
  - Releasing time + Processing time with doctor + Scalar + 10* Rand ()*

* Rand () has expected values which vary with criticality of each type of patient and range is between +/-5

Scheduling Problem Setup – $J_2 | R_j | \sum U_j$

- Modeled after a Job Shop model
  - 5 Job types

Job 1 – criticality 0 (cleared) – 10%	**Job 2 – criticality 1 (low) – 20%**
Job 3 – criticality 3 (Medium) – 20%	Job 4 – criticality 4 (High) – 30%
Job 5 – criticality 5 (Critical) – 20%	

  - 2 Machines:
    - triage – Machine 1
    - treatment – Machine 2
  - Death Time Metric:

	50 % chance	100% chance
Job 1	**Infinity**	**Infinity**
Job 2	Rj + A90 + P22	Rj + A180 + P22
Job 3	Rj + A36 + P23	Rj + A180 + P22
Job4	Rj + A18 + P24	Rj + A36 + P24
Job 5	Rj + A9 + P25	Rj + A18 + P25

R_j: Patients arrival time
U_j: Number of dead patients from the system
A9: random variable constant with expected value of 9, ranging from +/- 5.

Proposed Algorithms

- First In First Out :
  - Prioritize the patients with earliest arrival time
  - Process earliest arrived patient when machine becomes available
  - Same for both nurse queue and doctor queue (sorted on arrival time)

- Highest Criticality First / EDD:
  - Assign priority to the patients with the highest criticality
  - Higher criticality is associated with earliest time in death metrics, both for 50% and 100%

Job 2	85 + 10*rand()	175 + 10*rand()
Job 3	31 + 10*rand()	67 + 10* rand()
Job 4	13 + 10*rand()	31 + 10*rand()
Job 5	4 + 10*rand()	23 + 10*rand()

- Advanced Expected Death EDD:
  - Comparing completion time with death metrics for patients in queue
  - Discard the patients with 100% certainty to die regardless received treatment or not, then advance the process by taking the next patient in queue
  - Feasible Lower Bound

FIFO Model & Output Analysis

- Store patients in queues for doctor and nurse
- Sort doctor queue based on the arrival time
  - [Y,I] = sort(queueD(1,:),'descend');
    % queueD stores patients waiting to be treated by doctor
  - Check if the patient is dead before service starts

```
if (serviceD1(3,Ncustomers+1)>queueD(4,1))          %If job is dead before service
        serviceD1(2,NCustomers+1) = 0.001;            %set dead jobs as cleared
        serviceD1(3,NCustomers+1) = serviceD1(1,NCustomers+1) + serviceD1(2,NCustomers+1);
                                                      %end time
        NDeath = NDeath+1;
end
```

- Take the next on sorted queue when additional machine becomes available

Result: (20 exits in each simulation; 1000 simulations)

Number of 100% Death	Number of 50% Death	Total Number Completed	Make Span (minutes)
7.08	2.12	20	299.55

Highest Criticality First/EDD

Similar set up as the FIFO except when queueD >0 & doctor or nurse becomes available, use a different sorting schema

```
if (NextDoctorAvailable == 1)              % if doctor 1 is the next available doctor
 ...
     EDDQueueD = EDDQueueD - 1;
     [Y,I] = sort(queueD(1,:),'ascend');          % sort by criticality
     queueD = queueD(:,I);
     [Y,I] = sort(queueD(2,:),'descend');   % sort by arrival time
     queueD = queueD(:,I);
```

Result: (20 exits in each simulation; 1000 simulations)

Number of 100% Death	Number of 50% Death	Total Number Completed	Make Span (minutes)
7.28	2.08	20	299.83

Advanced EDD Output Analysis

```
% check is job dead before arrival
if serviceD1(1,Qcustomers+1)>queueD(4,1)        %If job is dead before service
    serviceD1(2,QCustomers+1) = 0.001;          %set dead jobs as cleared
    serviceD1(3,QCustomers+1) = serviceD1(1,QCustomers+1) + serviceD1(2,QCustomers+1);   %end time
    NDeath = NDeath+1;
end

% check is the job expected to die before service end
if serviceD1(3,QCustomers+1)>queueD(4,1)
    serviceD1(2,QCustomers+1) = 0.001;          %set dead jobs as cleared
    serviceD1(3,QCustomers+1) = serviceD1(1,QCustomers+1) + serviceD1(2,QCustomers+1);
                                                %end time
    NDeath = NDeath+1;
end
```

Result: (20 exits in each simulation; 1000 simulations)

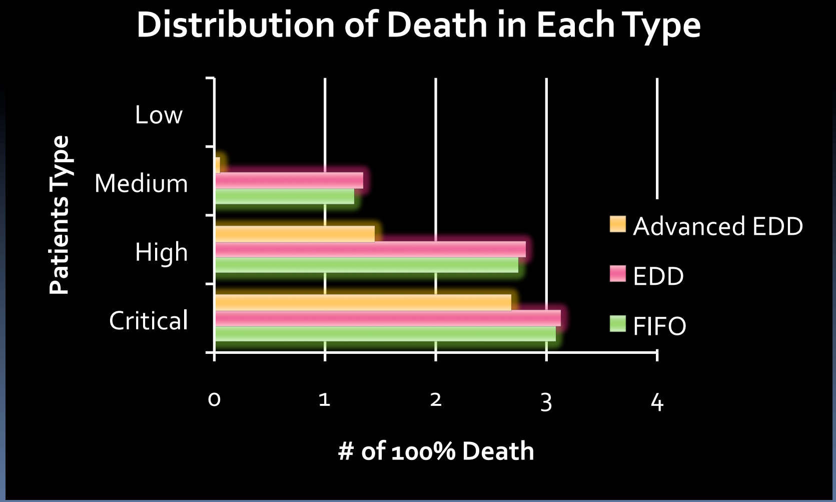
Number of 100% Death	Number of 50% Death	Total Number Completed	Make Span (minutes)
4.19	3.07	20	272.32

Results & Conclusions

	Number of 100% Death	Number of 50% Death	Make Span (minutes)
FIFO	7.08	2.12	299.55
EDD	7.28	2.08	299.83
Advanced EDD	4.19	3.07	272.32

- Advanced Expected Death EDD output can be used as a feasible LB
- EDD is not necessarily better than FIFO
- Make Span from FIFO and EDD are very close to each other
- EDD can result in large queues
  - i.e. for a simulation with 20 jobs, EDD can end up having >100 people in queue

- Advanced EDD has the least number of death in every type
- In all three algorithms, most death happens in "Critical"
- EDD yields higher number of death in all three types of patients



Also keep in mind … sanity check

- Advanced EDD = Let patients die before being seen by doctors or nurses?
  - How practical that really is?
- Who determine the “criticality” of the patients?
  - Possibilities of making mistake on assigning “criticality” to patients?
  - How do we know before patients going through arbitrage?
- Advanced EDD discard 100% certainty death, what about 50% death?
- Offline Scheduling
  - Know the arrival of each patients then schedule the jobs
  - Will this be a better LB?

Thank you!