Chap 11: Hierarchical Models

Objective: to avoid large models so as to improve solution efficiency.

Ex1:

upper-level model
(a reliability block diagram)

Lower-level model for a bridge
(a reliability graph)
Partial sharpe code shown

\[
\begin{align*}
\text{relgraph} & \quad \text{rbridge (v1, v2, v3, v4, v5)} \\
\text{w} & \quad \text{x} \quad \exp (v1) \\
\text{x} & \quad \text{z} \quad \exp (v2) \\
\text{w} & \quad \text{y} \quad \exp (v4) \\
\text{y} & \quad \text{z} \quad \exp (v5) \\
\text{bidirect} & \\
\text{x} & \quad \text{y} \quad \exp (v3) \\
\text{end} & \\
\text{block} & \quad \text{rel-in-block} \\
\text{comp} & \quad 11 \quad \exp (u11) \\
\text{comp} & \quad 12 \quad \exp (u12) \\
\text{comp} & \quad 13 \quad \exp (u13) \\
\text{comp} & \quad 14 \quad \exp (u14) \\
\text{comp} & \quad 15 \quad \exp (u15) \\
\text{comp} & \quad \text{bridge1} \quad \text{cdf (rbridge; u1, u2, u3, u4, u5)} \\
\text{comp} & \quad \text{bridge2} \quad \text{cdf (rbridge; u6, u7, u8, u9, u10)} \\
\text{parallel} & \quad \text{C} \quad 12 \quad 13 \\
\text{series} & \quad \text{D} \quad \text{bridge1} \quad 11 \quad \text{C} \\
\text{series} & \quad \text{E} \quad 14 \quad \text{bridge2} \quad 15 \\
\text{parallel} & \quad \text{top} \quad \text{D} \quad \text{E} \\
\text{end} & \\
\text{eval (rel-in-block)} & \quad 0 \quad 50000 \quad 500 \\
\text{end} & \\
\end{align*}
\]
Ex2: A queuing model with resource constraints

\[ P_{277} \]

# of running jobs in the central server is limited to \( n < M \)

\[ \text{Within the dashed line is the central server system} \]

Not in product-form because of resource limitations causing input flow ≠ output flow

\[ \text{In product-form: both servers can be evaluated independently} \]

\[ X(i) = \begin{cases} X(i), & \text{if } i = 1,2,\ldots,n \\ X(n), & \text{if } i > n \end{cases} \]
* low-level model

```
pfqn inner(n)
CPU  disk1  P1
CPU  disk2  P2
CPU  CPU    1-P1,P2
disk1 CPU    1
disk2 CPU    1
end
CPU  fcfs  1000/20
disk1 fcfs  1000/30
disk2 fcfs  1000/42.9
end
chain1 n
end
```

* high-level model

```
fqn outer(M)
term  is  1/25
central lds  X(1), X(2), X(3), X(4)
end
chain1 M
end
```

* define function for lds throughput X(n)

```
func X(n)  tput(inner,CPU;n)*(1- P1-P2)
```

* can also be obtained as

```
(1000/20 * util(inner,CPU;n))*(1- P1-P2)
```

* by Little’s Law, i.e., \( x_{CPU} = \frac{\mu_{CPU}}{\rho_{CPU}} \)

```
bind
P1 0.667
P2 0.233
end
```

* reporting each terminal user’s response time in

* the central system as the number of users (M)

* increases

```
loop i, 0, 4, 1
  expr 5*(2^i)
  expr rtime(outer, central; 5*(2^i))
end
end
Ex3: A queuing model with job priorities

Two classes of jobs: 1 & 2

- Low priority
- High priority at the CPU only

- $M_1 = 3$ & $M_2 = 4$
- $\lambda_1 = 1/12$; $\lambda_2 = 1/7$
- $P_0(\text{class 1}) = 1/15$
- $P_0(\text{class 2}) = 1/31$
- $P_1(\text{class 1}) = 8/15$
- $P_1(\text{class 2}) = 5/31$
- $P_2(\text{class 1}) = 5/15$
- $P_2(\text{class 2}) = 15/31$
- $P_3(\text{class 1}) = 1/15$
- $P_3(\text{class 2}) = 10/31$

Service demand:
- Class 1: 0.1 sec.
- Class 2: 0.06 sec.
- (Class 2 has higher priority at CPU)

Diagram:
- CPU
- Disks 1, 2, 3:
  - Disk 1: 0.03 sec.
  - Disk 2: 0.03 sec.
  - Disk 3: 0.03 sec.
* performance measures of interest: response time & queue length at CPU.
* not in product-form because of priority scheduling.

Approximation solution: suppose $u_2$ is the utilization of the CPU dedicated to class 2 jobs. Then the CPU service rate for class 1 jobs is slowed down by a factor of $(1-u_2)$

* we don’t know $u_2$ since it is an output, but we need it as an input for class 1 jobs.

\[ \therefore \text{use iterative technique} \]

Create two CPUs, one for class 1 & the other for class 2, with the CPU service rate to class 1 jobs reduced by a factor of $(1-u_2)$
Sharpe code (see p. 285, text)

\[
\text{mpfqn iter (M1, M2, u2)}
\]

* chain 1 for class 1 jobs

chain 1

\[
\begin{align*}
\text{CPU1} & \quad \text{disk1} & \quad 8/15 \\
\text{CPU1} & \quad \text{disk2} & \quad 5/15 \\
\text{CPU1} & \quad \text{disk3} & \quad 1/15 \\
\text{CPU1} & \quad \text{terminals} & \quad 1/15 \\
\text{disk1} & \quad \text{CPU1} & \quad 1 \\
\text{disk2} & \quad \text{CPU1} & \quad 1 \\
\text{disk3} & \quad \text{CPU1} & \quad 1 \\
\end{align*}
\]

end

* chain 2 for class 2 jobs

chain 2

\[
\begin{align*}
\text{CPU2} & \quad \text{disk1} & \quad 5/31 \\
\text{CPU2} & \quad \text{disk2} & \quad 15/31 \\
\text{CPU2} & \quad \text{disk3} & \quad 10/31 \\
\text{CPU2} & \quad \text{terminals} & \quad 1/31 \\
\text{disk1} & \quad \text{CPU2} & \quad 1 \\
\text{disk2} & \quad \text{CPU2} & \quad 1 \\
\text{disk3} & \quad \text{CPU2} & \quad 1 \\
\end{align*}
\]

end

* Section 2: server types

\[
\begin{align*}
\text{CPU1} & \quad \text{fcfs} & \quad (1-u_2)*1/0.1 \\
\text{CPU2} & \quad \text{fcfs} & \quad 1/0.06 \\
\text{disk1} & \quad \text{fcfs} & \quad 1/0.03 \\
\text{disk2} & \quad \text{fcfs} & \quad 1/0.03 \\
\text{disk3} & \quad \text{fcfs} & \quad 1/0.03 \\
\text{terminals} & \quad \text{fcfs} & \quad 1/0.03 \\
\end{align*}
\]

Service rate of class 1 jobs is reduced by a factor of \((1-u_2)\)

* Section 3: number of jobs per class

\[
\begin{align*}
1 & \quad \text{M1} \\
2 & \quad \text{M2} \\
\end{align*}
\]
we don’t know what the initial value of $u_2$ is, 
so make a guess $u_2=0$ initially

bind $u_2$ mutil (iter, CPU2, 2; 3, 4, 0)

parameters for M1, M2, & $u_2$

system name station name chain 2

continue this for a sufficient # of iterations 
until $u_2$ converges ⇒ try 5 times

loop i, 1, 5, 1

bind $u_2$ mutil (iter, CPU2, 2; 3, 4, $u_2$ )

end

outputs are:

* i=1 $u_2 \leftarrow 0.659839$
* i=2 $u_2 \leftarrow 0.659838$
* i=3 $u_2 \leftarrow 0.659838$

(continued after 3 iterations)

try starting $u_2$ with another initial value, 
say $u_2 =0.9$

bind $u_2$ 0.9

loop 1, 1, 5, 1

bind $u_2$ mutil (iter, CPU2, 2; 3, 4, $u_2$ )

end

M1=3; M2=4 & $u_2$ is equal to the $u_2$ in the previous iteration

* outputs are

$u_2$ is also converged in 3 iterations

printing response time & queue size

expr mrttime (iter, CPU1, 1; 3, 4, $u_2$ )
expr mrttime (iter, CPU2, 2; 3, 4, $u_2$ )
mqlength (iter, CPU1, 1; 3, 4, $u_2$ )
mqlength (iter, CPU2, 2; 3, 4, $u_2$ )

outputs are

$R_{1, CPU}=0.47534$
to be compared with

$R_{2, CPU}=0.10511$

$n_{1, CPU} = 1.0911$

$n_{2, CPU} = 1.1559$

end
Ex4: M/M/1/k queue with server failure & repair

\[ \begin{align*} P.233, \text{ text & p.294} & \quad \gamma: \text{ failure rate} \quad \lambda: \text{ job arrival rate} \\
& \quad \tau: \text{ repair rate} \quad \mu: \text{ job service rate} \\
\end{align*} \]

1-level model

M/M/1/10

State representation \((a, b)\)

\[ \begin{align*}
\text{# of jobs} & \quad \{ 1 \text{ alive} \quad 0 \text{ failed} \} \\
\text{prob \{idle server\}} & = \text{prob}_{(0,0)} + \text{prob}_{(0,1)} \\
\text{rejection probability} & = \text{prob}_{(10,0)} + \text{prob}_{(10,1)}
\end{align*} \]
Two-level model

observation: job arrivals/services are much faster than server failures/repairs

\[ \therefore \text{ the assumption below is justified:} \]

“the set of states 0,1 1,1 .............. 9,1 10,1 whose transitions are job arrivals and departures will reach equilibrium between the times when a failure/repair occurs.”

\[ \Rightarrow \text{isolate out the fast recurrent set of states from the 1-level model, analyze it for steady-state probabilities & replace it by a single state in the original model.} \]
High-level:

Low-level:

\[
\begin{align*}
\text{Prob\{idle server\}} &= \text{prob(high-model, } 0,0) \\
&\quad + \text{prob(high-model, } 1) \times \text{prob(low-model, } 0,1) \\
\text{Rejection prob} &= \text{prob(high-model, } 10,0) \\
&\quad + \text{prob(high-model, } 1) \times \text{prob(low-model, } 10,1)
\end{align*}
\]