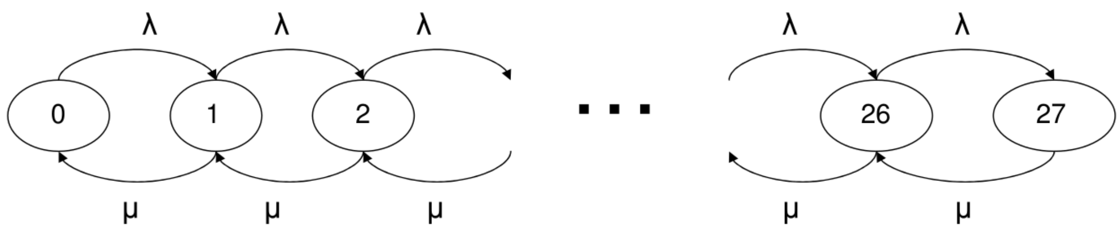
Math 420/620 Project

A mechanic working for a heavy equipment repair facility is responsible for the repair and maintenance of forklift trucks. When forklifts break down they are taken to the repair facility and serviced in the order of their arrival. Assume rate at which forklifts break down is λ per month and the rate they can be repaired is μ per month. Since there is space for 27 forklifts at the facility, the number of forklifts in repair at time t can be represented by

$$X_t \in \{0, 1, 2, \ldots, 27\}.$$

The only possible transitions are from $X_t = i$ to $X_t = i + 1$ or $i - 1$. Thus, the transition rate diagram looks like



and the probability distribution $P(t) = \big(P_0(t), P_1(t), P_2(t), \ldots, P_{27}(t)\big)$ of the corresponding Markov process satisfies

$$\frac{dP}{dt} = PA \qquad \text{where} \qquad A = \begin{bmatrix} -\lambda & \lambda & & & 0 \\ \mu & -\mu-\lambda & \ddots & & \\ & \mu & \ddots & \lambda & \\ & & \ddots & -\mu-\lambda & \lambda \\ 0 & & & \mu & -\mu \end{bmatrix}.$$

Note that $A \in \mathbf{R}^{28\times 28}$ and $P \in \mathbf{R}^{1\times 28}$ is a row vector.

Note that answers will be different with the version because each project version has different values of λ and μ. As a result we create a list of those indexed by version number here.

In [1]:
```
Ls=[4.9,4.8,3.7,4.8,4.9,5.2,3.6,5.5,4.2]
Ms=[7.4,7.7,5.6,6.8,7.3,8.3,7.0,7.4,6.1]
Ps=[1,2,4,5,6,7,8,9,10]
Is=1:length(Ps)
using PrettyTables
T1=[Ps Ls Ms]
pretty_table(T1,
```

```
    formatters=[fmt__printf("%3d",[1]),fmt__printf("%.1f",[2,3])],
    column_labels=["version","lambda","mu"])
```

version	lambda	mu
1	4.9	7.4
2	4.8	7.7
4	3.7	5.6
5	4.8	6.8
6	4.9	7.3
7	5.2	8.3
8	3.6	7.0
9	5.5	7.4
10	4.2	6.1

(i) Solve $PA = 0$ where $P_i \geq 0$ and $\sum_{i=0}^{27} P_i = 1$ to obtain the steady-state distribution $P = (P_0, P_1, P_2, \ldots, P_{27})$ such that $P_i = \mathbf{P}(X_t = i)$ in the limit.

In [2]:
```
using LinearAlgebra
# Given lambda and mu make the tridiagonal matrix
function mkA(lambda,mu)
    a1=mu*ones(27)
    a3=lambda*ones(27)
    a2=[-lambda,-(mu+lambda)ones(26)...,-mu]
    return Tridiagonal(a1,a2,a3)
end
```

Out[2]:
```
mkA (generic function with 1 method)
```

In [3]:
```
# Example matrix for version 1.  Set COLUMNS so no wrapping later.
ENV["COLUMNS"]=72
A=mkA(Ls[1],Ms[1])
```

Out[3]:
```
28×28 Tridiagonal{Float64, Vector{Float64}}:
 -4.9    4.9      ⋅       ⋅       ⋅   …    ⋅       ⋅       ⋅      ⋅
  7.4  -12.3    4.9       ⋅       ⋅        ⋅       ⋅       ⋅      ⋅
   ⋅     7.4  -12.3     4.9       ⋅        ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅     7.4   -12.3     4.9        ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅      7.4   -12.3        ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅      7.4   …    ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅    …    ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
  ⋮                                    ⋱                  ⋮
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅    …    ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅        4.9      ⋅       ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅      -12.3     4.9      ⋅      ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅    …   7.4   -12.3     4.9     ⋅
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅      7.4   -12.3    4.9
   ⋅      ⋅      ⋅       ⋅       ⋅         ⋅       ⋅      7.4   -7.4
```

In [4]:
```
# Find the steady state
function mkP0(A)
    K28=eigvecs(A')[:,28]
    return K28/sum(K28)
end
```

Out[4]:
```
mkP0 (generic function with 1 method)
```

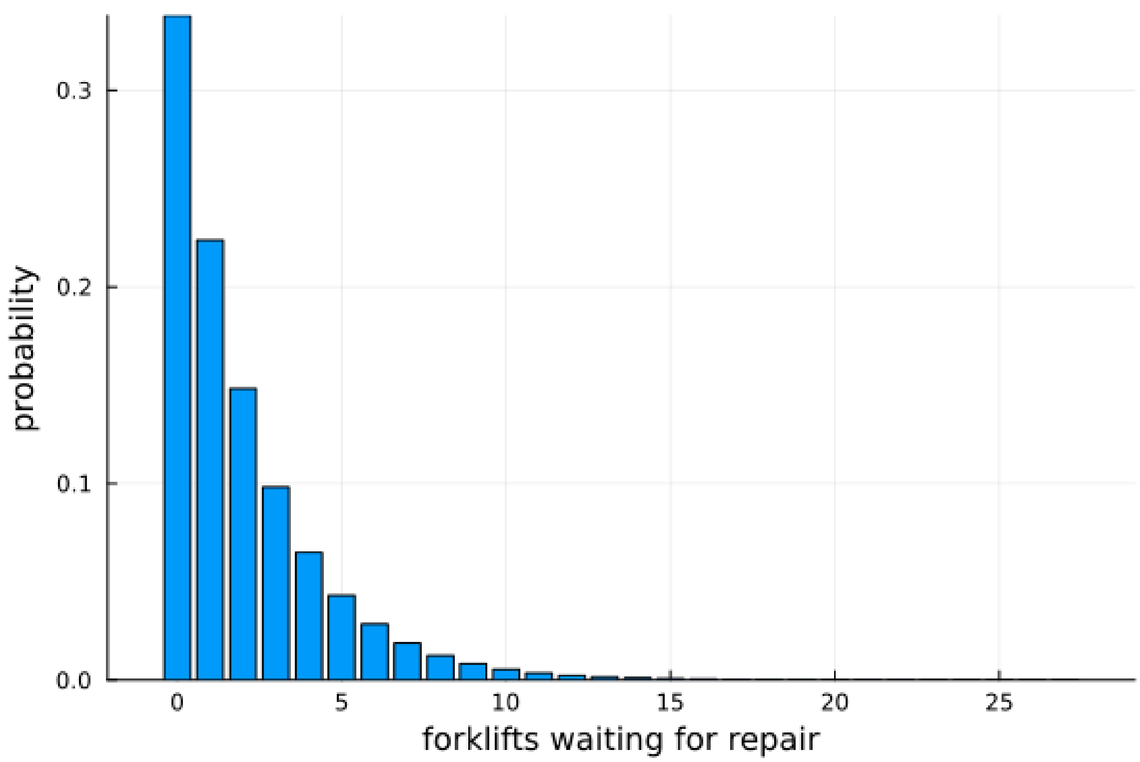
In [5]:
```
P0=mkP0(A)
T2=[0:13 P0[1:14] 14:27 P0[15:28]]
pretty_table(T2,
    formatters=[fmt__printf("%8d",[1,3]),fmt__printf("%.15f",[2,4])],
    column_labels=["n","P0(n)","n","P0(n)"])
```

n	P0(n)	n	P0(n)
0	0.337841116635315	14	0.001052479263803
1	0.223705604258519	15	0.000696911944950
2	0.148129386603614	16	0.000461468720305
3	0.098085674913204	17	0.000305567125607
4	0.064948622577662	18	0.000202334988578
5	0.043006520355479	19	0.000133978573518
6	0.028477290505655	20	0.000088715541924
7	0.018856584253744	21	0.000058744075058
8	0.012486116600453	22	0.000038898103754
9	0.008267833965164	23	0.000025756852486
10	0.005474646814771	24	0.000017055213133
11	0.003625103971943	25	0.000011293316804
12	0.002400406684124	26	0.000007478007073
13	0.001589458480028	27	0.000004951653332

Remark that the above table and following histogram are for version 1.

In [6]:
```
using Plots
bar(0:27,P0,legend=:false,
    xlabel="forklifts waiting for repair",ylabel="probability")
```

Out[6]:



(ii) Compute percentage $\mathbf{P}(X_t = 0)$ of time the mechanic does not have work.

Assuming no forklifts are broken at time $t = 0$ the For convenience convert everything to column vectors by writing $w = P^T$ and $B = A^T$. The differential equation involving the evolution of w is $dw/dt = Bw$.

Suppose K_i are the eigenvectors of B with corresponding eigenvalues α_i. Then the general solution is of the form

$$w(t) = c_1 K_1 e^{\alpha_1 t} + c_2 K_2 e^{\alpha_2 t} + \cdots + c_{28} K_{28} e^{\alpha_{28} t}.$$

Assume no forklifts are in for repair at time $t = 0$. Therefore

$$w(0) = (1, 0, \ldots, 0) = e_1.$$

In [7]:
```
B=A'
K=eigvecs(B)
alpha=eigvals(B)
e1=I(28)[:,1]
c=K\e1
w(t)=sum(c[i]*K[:,i]*exp(alpha[i]*t) for i=1:28)
```

Out[7]:
```
w (generic function with 1 method)
```

In [8]:
```
# w0(t) is the probability the mechanic doesn't have work at time t
w0(t)=w(t)[1]
```

Out[8]:
```
w0 (generic function with 1 method)
```

In [9]:
```
# Up to rounding no work to start with
w0(0)
```

Out[9]:
```
0.9999999999999998
```

In [10]:
```
# The equilibrium probablity there is no work
P0[1]
```

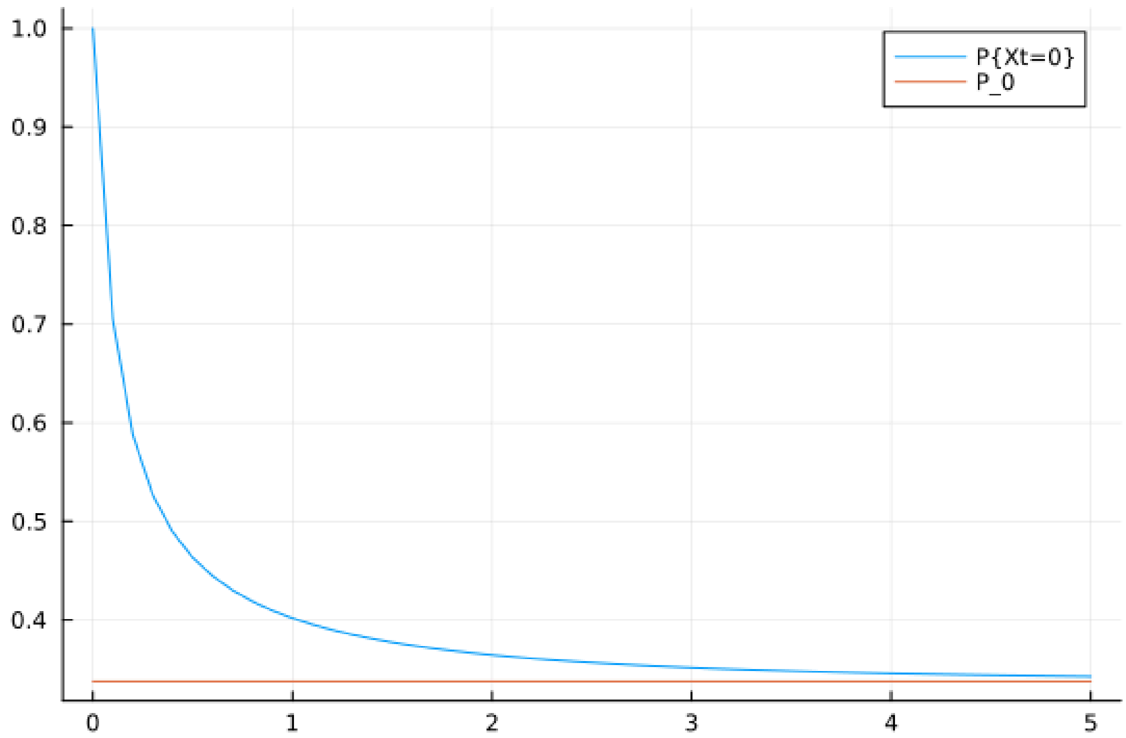
Out[10]:
```
0.33784111663531485
```

We know w converges to the equilibrium state. Thus,

$$w(t) \to P_0 \approx 0.33784 \qquad \text{as} \qquad t \to \infty.$$

In [11]:
```
plot(w0,0:0.1:5,label="P{Xt=0}")
plot!(t->P0[1],0:0.1:5,label="P_0")
```

Out[11]:



The above graph showing the evoltion of $w(t)$ is for version 1. We loop through all the different values of λ and μ to obtain the probability the mechanic doesn't have work for each of the project versions.

In [12]:

```
P0s=[mkP0(mkA(Ls[i],Ms[i]))[1] for i=Is]
T3=[Ps Ls Ms P0s]
pretty_table(T3,
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.2f",[2,3]),fmt__printf("%.15f",[4])],
    column_labels=["version","lambda","mu","P(Xt=0)"])
```

version	lambda	mu	P(Xt=0)
1	4.90	7.40	0.337841116635315
2	4.80	7.70	0.376624051030879
4	3.70	5.60	0.339288811370224
5	4.80	6.80	0.294134749236972
6	4.90	7.30	0.328771793586915
7	5.20	8.30	0.373494745373329
8	3.60	7.00	0.485714289696327
9	5.50	7.40	0.256820041068969
10	4.20	6.10	0.311484429307606

(iii) Compute the expected queue length $\mathbf{E}[X_t]$ of forklifts in the repair facility.

By definition

$$\mathbf{E}[X_t] = \sum_{n=0}^{27} n\mathbf{P}(X_t = n).$$

For large t this reflects the equilibrium state. Thus,

$$\mathbf{E}[X_t] \approx \lim_{t\to\infty} \mathbf{E}[X_t] = \sum_{n=0}^{27} nP_n.$$

In [13]:
```
function getE(i)
    P=mkP0(mkA(Ls[i],Ms[i]))
    # remark the P[n] is stored off by one
    return sum(n*P[n+1] for n=0:27)
end
Es=getE.(Is)
T4=[Ps Ls Ms P0s Es]
pretty_table(T4,
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.2f",[2,3]),fmt__printf("%.8f",[4,5])],
    column_labels=["version","lambda","mu","P(Xt=0)","E[Xt]"])
```

version	lambda	mu	P(Xt=0)	E[Xt]
1	4.90	7.40	0.33784112	1.95972825
2	4.80	7.70	0.37662405	1.65512228
4	3.70	5.60	0.33928881	1.94711283
5	4.80	6.80	0.29413475	2.39837187
6	4.90	7.30	0.32877179	2.04126891
7	5.20	8.30	0.37349475	1.67736167
8	3.60	7.00	0.48571429	1.05882330
9	5.50	7.40	0.25682004	2.88783552
10	4.20	6.10	0.31148443	2.20971551

(iii) What percentage of the time are 5 or more forklifts in the facility?

To compute $P(X_t \geq 5)$ again assume t is so the probabilities reflect the equilibrium state. Thus

$$P(X_t \geq 5) \approx \sum_{n=5}^{27} P_n.$$

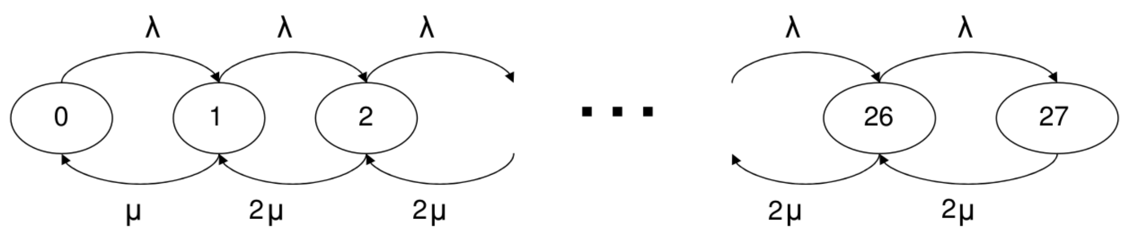
In [14]:
```
function getP5(i)
    P=mkP0(mkA(Ls[i],Ms[i]))
    # remark the P[n] is stored off by one
    return sum(P[n+1] for n=5:27)
end
P5s=getP5.(Is)
T5=[Ps Ls Ms P5s]
pretty_table(T5,
    formatters=[fmt__printf("%3d",[1]),
```

```
        fmt__printf("%.2f",[2,3]),fmt__printf("%.15f",[4])],
    column_labels=["version","lambda","mu","P(Xt>=5)"])
```

version	lambda	mu	P(Xt>=5)
1	4.90	7.40	0.127289595011686
2	4.80	7.70	0.094133688738398
4	3.70	5.60	0.125904416258132
5	4.80	6.80	0.175203494390092
6	4.90	7.30	0.136247013199614
7	5.20	8.30	0.096520126235555
8	3.60	7.00	0.035976773199740
9	5.50	7.40	0.226615588095287
10	4.20	6.10	0.154713629597006

(iv) Now suppose that a second mechanic is called in whenever two or more forklifts need repair. State the new rate matrix $\tilde{A}$ and find the corresponding steady-state distribution $\mathbf{P}(\tilde{X}_t = i)$ of the number of forklifts in the facility.

The new rate diagram looks like



Therefore, the corresponding matrix is

$$\tilde{A} = \begin{bmatrix} -\lambda & \lambda & & & & 0 \\ \mu & -\mu-\lambda & \lambda & & & \\ & 2\mu & -2\mu-\lambda & \ddots & & \\ & & 2\mu & \ddots & \lambda & \\ & & & \ddots & -2\mu-\lambda & \lambda \\ 0 & & & & 2\mu & -2\mu \end{bmatrix}.$$

In [15]:
```
function mktA(lambda,mu)
    a1=[mu,2*mu*ones(Int,26)...]
    a3=lambda*ones(Int,27)
    a2=[-lambda,-(mu+lambda),-(2*mu+lambda)ones(Int,25)...,-2*mu]
    return Tridiagonal(a1,a2,a3)
end
```

Out[15]: mktA (generic function with 1 method)

In [16]:
```
# check it is correct using Symbolics
using Symbolics
@variables λ,μ
mktA(λ,μ)
```

Out[16]:
```
28×28 Tridiagonal{Num, Vector{Num}}:
 -λ        λ         ⋅         ⋅   …       ⋅         ⋅         ⋅    ⋅
  μ  -λ - μ         λ         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅       2μ  -λ - 2μ         λ           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅        2μ  -λ - 2μ           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅        2μ           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅   …       ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅   …       ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋮                                 ⋱                  ⋮
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅   …       ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅           λ         ⋅         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅     -λ - 2μ         λ         ⋅    ⋅
  ⋅        ⋅         ⋅         ⋅   …      2μ  -λ - 2μ         λ    ⋅
  ⋅        ⋅         ⋅         ⋅           ⋅        2μ  -λ - 2μ    λ
  ⋅        ⋅         ⋅         ⋅           ⋅         ⋅        2μ  -2μ
```

In [17]:
```
tP0=mkP0(mktA(Ls[1],Ms[1]))
T6=[0:13 tP0[1:14] 14:27 tP0[15:28]]
pretty_table(T6,
    formatters=[fmt__printf("%8d",[1,3]),fmt__printf("%.15f",[2,4])],
    column_labels=["n","tP0(n)","n","tP0(n)"])
```

n	tP0(n)	n	tP0(n)
0	0.502538071066017	14	0.000000191108529
1	0.332761695705876	15	0.000000063272418
2	0.110171101956675	16	0.000000020948301
3	0.036475567539710	17	0.000000006935586
4	0.012076370334093	18	0.000000002296241
5	0.003998257745747	19	0.000000000760242
6	0.001323747496903	20	0.000000000251702
7	0.000438267752353	21	0.000000000083334
8	0.000145102161252	22	0.000000000027590
9	0.000048040580414	23	0.000000000009135
10	0.000015905327299	24	0.000000000003024
11	0.000005265952957	25	0.000000000001002
12	0.000001743457398	26	0.000000000000331
13	0.000000577225760	27	0.000000000000110

Remark the above table is only for version 1.

(vi) Use the results of part (v) to calculate the expected number $\mathbf{E}[\tilde{X}_t]$ of forklifts in the repair facility, the probability that the first mechanic is busy, the probability that the second mechanic is called in and the probability that 5 or more forklifts are in the facility.

For large t the expected number of forklifts in the facility is given by

$$\mathbf{E}[\tilde{X}_t] \approx \sum_{n=0}^{27} n\tilde{P}_n.$$

The probability that the first mechanic is busy is given by

$$\mathbf{P}(\tilde{X}_t \geq 1) = 1 - \mathbf{P}(\tilde{X}_t = 0) \approx 1 - \tilde{P}_0$$

The probability that the second mechanic is called in is given by

$$\mathbf{P}(\tilde{X}_t \geq 2) = 1 - \mathbf{P}(\tilde{X}_t < 2) \approx 1 - \tilde{P}_0 - \tilde{P}_1$$

and the probability that 5 or more forklifts are in the facility is

$$\mathbf{P}(\tilde{X}_t \geq 5) \approx \sum_{n=5}^{27} \tilde{P}_n.$$

In [18]:
```
function gettE(i)
    tP=mkP0(mktA(Ls[i],Ms[i]))
    # remark the P[n] is stored off by one
    return sum(n*tP[n+1] for n=0:27)
end
tEs=gettE.(Is)
```

```
tP0s=[mkP0(mktA(Ls[i],Ms[i]))[1] for i=Is]
tP2s=[1-sum(mkP0(mktA(Ls[i],Ms[i]))[1:2]) for i=Is]
tP5s=[sum(mkP0(mktA(Ls[i],Ms[i]))[6:28]) for i=Is]
T7=[Ps Ls Ms tEs 1.0.-tP0s tP2s tP5s]
pretty_table(T7,
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.2f",[2,3]),
        fmt__printf("%.6f",[4,5,6,7])],
    column_labels=["version","lambda","mu","E[tXt]",
        "P(tXt>=1)","P(tXt>=2)","P(tXt>=5)"])
```

version	lambda	mu	E[tXt]	P(tXt>=1)	P(tXt>=2)	P(tXt>=5)
1	4.90	7.40	0.743680	0.497462	0.164700	0.005977
2	4.80	7.70	0.690454	0.475248	0.148129	0.004485
4	3.70	5.60	0.741655	0.496644	0.164070	0.005915
5	4.80	6.80	0.806324	0.521739	0.184143	0.008096
6	4.90	7.30	0.756437	0.502564	0.168669	0.006376
7	5.20	8.30	0.694672	0.477064	0.149442	0.004594
8	3.60	7.00	0.550699	0.409091	0.105195	0.001789
9	5.50	7.40	0.862334	0.541872	0.201371	0.010335
10	4.20	6.10	0.781098	0.512195	0.176329	0.007194

(vii) Compare the results in (vi) to the results with only one mechanic.

In [19]:

```
P2s=[1-sum(mkP0(mkA(Ls[i],Ms[i]))[1:2]) for i=Is]
T8t=[[Ps[i],Ls[i],Ms[i],[Es[i],tEs[i]],
    [1-P0s[i],1-tP0s[i]],[P2s[i],tP2s[i]],[P5s[i],tP5s[i]]]
    for i=Is]
T8=reduce(vcat,T8t')
```

Out[19]:

```
9×7 Matrix{Any}:
  1  4.9  7.4  [1.95973 0.74368]   …  [0.12729 0.00597719]
  2  4.8  7.7  [1.65512 0.690454]     [0.0941337 0.00448541]
  4  3.7  5.6  [1.94711 0.741655]     [0.125904 0.00591535]
  5  4.8  6.8  [2.39837 0.806324]     [0.175203 0.00809586]
  6  4.9  7.3  [2.04127 0.756437]     [0.136247 0.00637624]
  7  5.2  8.3  [1.67736 0.694672]  …  [0.0965201 0.00459365]
  8  3.6  7.0  [1.05882 0.550699]     [0.0359768 0.00178862]
  9  5.5  7.4  [2.88784 0.862334]     [0.226616 0.0103348]
 10  4.2  6.1  [2.20972 0.781098]     [0.154714 0.00719438]
```

In [20]:

```
using Printf
printcell(v,_,_)=@sprintf("%.6f\n%.6f",v[1],v[2])
pretty_table(T8,
    line_breaks=true,
    table_format=TextTableFormat(horizontal_lines_at_data_rows=:all),
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.2f",[2,3]),(v,i,j)->j>3 ? printcell(v,i,j) : v],
    column_labels=["version","lambda","mu","E[Xt]",
        "P(Xt>=1)","P(Xt>=2)","P(Xt>=5)"],
    fit_table_in_display_vertically=false)
```

version	lambda	mu	E[Xt]	P(Xt>=1)	P(Xt>=2)	P(Xt>=5)
1	4.90	7.40	1.959728 0.743680	0.662159 0.497462	0.438453 0.164700	0.127290 0.005977
2	4.80	7.70	1.655122 0.690454	0.623376 0.475248	0.388597 0.148129	0.094134 0.004485
4	3.70	5.60	1.947113 0.741655	0.660711 0.496644	0.436538 0.164070	0.125904 0.005915
5	4.80	6.80	2.398372 0.806324	0.705865 0.521739	0.498241 0.184143	0.175203 0.008096
6	4.90	7.30	2.041269 0.756437	0.671228 0.502564	0.450546 0.168669	0.136247 0.006376
7	5.20	8.30	1.677362 0.694672	0.626505 0.477064	0.392509 0.149442	0.096520 0.004594
8	3.60	7.00	1.058823 0.550699	0.514286 0.409091	0.264490 0.105195	0.035977 0.001789
9	5.50	7.40	2.887836 0.862334	0.743180 0.541872	0.552300 0.201371	0.226616 0.010335
10	4.20	6.10	2.209716 0.781098	0.688516 0.512195	0.474051 0.176329	0.154714 0.007194

For each multi-line entry of the table of the form

1.959728 0.720465

the first line indicates the original statistic with only one mechanic available while the second line indicates the new statistic when the second mechanic is available.

For all project versions, adding a second mechanic reduces the expected repair queue length, decreases the percentage of time the first mechanic is working and decreases the percentage of time there are five or more forklifts waiting in the queue.

(viii) The cost of the second mechanic is $250 per day and the mechanic is only paid for the days worked. Another option is to lease replacement forklifts for the customers whose forklifts are in for repair but only during periods of backlog when two or more forklifts are in the facility. If the cost of replacements is $25 per day per forklift, which of the two plans is most cost effective?

Since a second mechanic only works when there is backlog, the expected cost of the second mechanic is

$$C_1 = 250\mathbf{P}(\tilde{X}_t \geq 2) \approx 250\tilde{P}_2$$

In [21]:
```
# Expected daily cost for second mechanic
C1(i)=250*tP2s[i]
C1(1)
```

Out[21]: 41.17505830702667

For leasing forklifts I assumed

- All customers whose forklifts are in the shop get a replacement forklift during times of backlog, including the forklift being repaired.

Note that it may have happened that the first forklift came into the shop at a time there was no backlog, so the mechanic immediately started repair and no replacement forklift provided. However, before the repair finished another forklift could break down. In this case repairs continue on the first forklift, but the first customer is now entitled to a replacement forklift. Curiously, this would encourage customers to bring in their broken forklifts as batches.

On the other hand, it may have happened that the forklift currently being repaired came into the shop when there was backlog but now that it's being repared there is no backlog. The moment repairs start, the previously obtained replacment forklift has to be returned.
In my opinion it would seem unfair a customer has to return the replacement forklift before getting their own back from the shop.

Our first goal is to compute the expected value of the queue length under the assumption of backlog.

$$\mathbf{E}[X_t|X_t \geq 2] = \sum_{n=2}^{27} nP(X_t = n|X_t \geq 2) = \sum_{n=2}^{27} \frac{nP(X_t = n)}{P(X_t \geq 2)}.$$

The expected cost is then

$$C_2 = 25\mathbf{E}[X_t|X_t \geq 2]P(X_t \geq 2) \approx \sum_{n=2}^{27} 25nP_n = 25(\mathbf{E}[X_t] - P_1).$$

In [22]:
```
# Expected daily cost for leased forklifts
C2(i)=25*(Es[i]-P0[2])
C2(1)
```

Out[22]: 43.40056622516525

In [23]:
```
C1s=C1.(Is)
C2s=C2.(Is)
Rs=[C1s[i]<C2s[i] ? "mechanic" : "lease" for i=Is]
T9=[Ps Ls Ms C1s C2s Rs]
pretty_table(T9,
    line_breaks=true,
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.1f",[2,3]),
        fmt__printf("%.2f",[4,5]),
        fmt__printf("%s",[6])],
    style=TextTableStyle(column_label=crayon"bold"),
    column_labels=[["version","lambda","mu",
            "second","leased","recommend"],
        ["","","","mechanic","forklifts",""]])
```

version	lambda	mu	second mechanic	leased forklifts	recommend
1	4.9	7.4	41.18	43.40	mechanic
2	4.8	7.7	37.03	35.79	lease
4	3.7	5.6	41.02	43.09	mechanic
5	4.8	6.8	46.04	54.37	mechanic
6	4.9	7.3	42.17	45.44	mechanic
7	5.2	8.3	37.36	36.34	lease
8	3.6	7.0	26.30	20.88	lease
9	5.5	7.4	50.34	66.60	mechanic
10	4.2	6.1	44.08	49.65	mechanic

Sometimes leasing is less expensive alternative sometimes not.

(ix) There is some uncertainty as to the actual cost of bringing in a second mechanic during periods of backlog. What is the minimum cost per day for a second mechanic that makes leasing a better alternative?

Since the cost of the second mechanic is just a multiple of their daily pay, we solve for constant β such that

$$\beta \mathbf{P}(\tilde{X}_n \geq 2) = 25C_2.$$

In [24]:
```
beta(i)=C2(i)/tP2s[i]
```

Out[24]: beta (generic function with 1 method)

In [25]:
```
Bs=[beta(i) for i=Is]
```

Out[25]:
```
9-element Vector{Float64}:
 263.51247581450775
 241.5826269315988
 262.60244507386216
 295.2411496314033
 269.3983100559957
 243.18096193828623
 198.46932889265483
 330.74842656464773
 281.5765748972829
```

In [26]:
```
T10=[Ps Ls Ms C1s C2s Bs]
pretty_table(T10,
    line_breaks=true,
    formatters=[fmt__printf("%3d",[1]),
        fmt__printf("%.1f",[2,3]),
        fmt__printf("%.2f",[4,5,6])],
    style=TextTableStyle(column_label=crayon"bold"),
    column_labels=[["version","lambda","mu",
            "second","leased","equivalent"],
        ["","","","mechanic","forklifts","daily pay"]])
```

version	lambda	mu	second mechanic	leased forklifts	equivalent daily pay
1	4.9	7.4	41.18	43.40	263.51
2	4.8	7.7	37.03	35.79	241.58
4	3.7	5.6	41.02	43.09	262.60
5	4.8	6.8	46.04	54.37	295.24
6	4.9	7.3	42.17	45.44	269.40
7	5.2	8.3	37.36	36.34	243.18
8	3.6	7.0	26.30	20.88	198.47
9	5.5	7.4	50.34	66.60	330.75
10	4.2	6.1	44.08	49.65	281.58

The equivalent daily pay for the second mechanic to compete with leasing replacement forklifts is listed in the last column. Sometimes it is greater than $250 and sometimes smaller depending on the rate of repair and breakdown of the forklifts.

In [ ]: