**Supplementary Material**: **Optimal Job Scheduling and Inspection of a Machine with Delayed Failure**

**Equation S1.** The recursive formula for minimal repair and replacement of the single machine at failure and inspection time considering *i* jobs in a sequence



S2. Detailed Heuristic algorithm

 The main scheme of heuristic algorithm adopted from (S. Wang and Liu 2013) is introduced in below: the idea is ﬁrst to construct a schedule according to Shortest Processing Time algorithm and then to make it feasible for the problem. The algorithm goes through  iterations. In iteration  of the algorithm, jobs are taken into consideration.

Step 1: Use the  rule to generate a job sequence.

Step 2: For  to length of (jobs) do;

Step 3: determine the inspection policy before each job, , , use the following rule;  represents the age of the machine after processing job  in the sequence and  shows the expected number of failures during processing of job ; where 



Step 4*:* If counter, go to step 5; otherwise, set  and go to Step 3.

Step 5: The resulting objective value is used as the expected makespan of the jobs.

Clearly, the minimum expected required time for processing each job is obtained when the machine is brand new. In the similar way the expected number of failures during processing of the jobs would be affected by the machine’s age. As the machine ages, the expected number of failures would be higher. We use this property in developing the heuristic algorithm.

S3. MATLAB code for expected makespan of one job

Data = xlsread('C:\data\Jobs processing times.xls');

% function expected makespan when there is only one job

function Expected Values=Expected\_One\_Job (t0,p,beta1,eta1,beta2,eta2,df)

 F=@(x)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(1-exp(-1.\*(((p+t0)/eta2).^beta2-((x+t0)/eta2).^beta2))); % Expected defect arrival time

 Probability\_failure\_less\_than\_p = integral(F,0,p);

 F = @(x,y)(x+y).\*(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2)); % Expected failure arrival time

 Probability \_failure\_less\_than\_p = integral2(F,0,p,0,@(x)p-x);

 ExpectedValues (1)= p+(df\*PZ\_less\_than\_p+ExpectedZ\_less\_than\_p)/(1-PZ\_less\_than\_p);

 ExpectedValues(2)= Probability \_failure\_less\_than\_p /(1- Probability \_failure\_less\_than\_p);

 ExpectedValues (3)= 0;

end

S4. MATLAB code for expected makespan of more than one job

% The input parameters, jobs processing times, defect arrival and delay time parameters, downtime parameters are exported from the excel file % The outputs which are the expected makespan, optimal job sequence and inspection policy is exported to the excel files

Data = xlsread ('C:\data\Jobs processing times.xls');

FunctionExpectedValues=MorethanOneJob(t0,a,b,ExpectedValuesMatrix1,ExpectedValuesMatrix2,JobsList,PMScheme,beta1,eta1,beta2,eta2,df,dp,dm,InspectionTime) % Expected value for more than one job in a sequence

 NumberofJobs=length(JobsList(:,1));

 ExpectedValues(1)=0;

 ExpectedValues(2)=0;

 ExpectedValues(3)=0;

 for i=1:NumberofJobs

 if i==1

 F=@(x)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(1-exp(-1.\*(((JobsList(i,2)+t0)/eta2).^beta2-((x+t0)/eta2).^beta2)));

 PZ\_less\_than\_p = integral(F,0,JobsList(i,2));

 F = @(x,y)(x+y).\*(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2));

 ExpectedZ\_less\_than\_p = integral2(F,0,JobsList(i,2),0,@(x)JobsList(i,2)-x);

 ExpectedValues(1)=df\*PZ\_less\_than\_p+ExpectedZ\_less\_than\_p;

 ExpectedValues(2)=PZ\_less\_than\_p;

 ExpectedValues(3)=0;

 else

 L2=sum(JobsList(1:i-1,2));

 U2=sum(JobsList(1:i,2));

 for j=1:i %Number of Jobs

 if (i==j)

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2)); % probability of failure and defect arrival time

 if j==1

 L1=0;

 U1=JobsList(1,2);

 else

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 end;

 int2 =integral2(F,L1,U1,0,@(x)U2-x);

 F4 = @(x,y)(x+y-sum(JobsList(1:i-1,2))).\*(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2)); % probability of failure and defect arrival time

 ExpectedTime=integral2(F4,L1,U1,0,@(x)U2-x);

 ExpectedValues(1)=ExpectedValues(1)+(df+ExpectedValuesMatrix1(1,i)+sum(JobsList(1:i-1,2)))\*int2+ExpectedTime;

 ExpectedValues(2)=ExpectedValues(2)+(1+ExpectedValuesMatrix1(2,i))\*int2;

 ExpectedValues(3)=ExpectedValues(3)+ExpectedValuesMatrix1(3,i)\*int2;

 else

 k=j+1;

 stillcheck=1;

 while k <= i && stillcheck==1

 if (i ~= j) && IndicatorLastoneOthersZero(PMScheme(j+1:k))==1

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2)); % probability of failure and defect arrival time

 if j==1

 L1=0;

 U1=JobsList(1,2);

 else

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 end;

 int2 =integral2(F,L1,U1,@(x)L2-x,@(x)U2-x);

 Prob=MinimalRepairProb(a,b,U1);

 ExpectedValues(1)=ExpectedValues(1)+(Prob\*(dm+ExpectedValuesMatrix2(1,k))+(1-Prob)\*(dp+ExpectedValuesMatrix1(1,k))+sum(JobsList(1:k-1,2)))\*int2;

 % ExpectedValues(2)=ExpectedValues(2)+ExpectedValuesMatrix2(2,k)\*int2;

 % ExpectedValues(3)=ExpectedValues(3)+(1+ExpectedValuesMatrix2(3,k))\*int2;

 ExpectedValues(2)=ExpectedValues(2)+(Prob\*ExpectedValuesMatrix2(2,k)+(1-Prob)\*ExpectedValuesMatrix1(2,k))\*int2;

 ExpectedValues(3)=ExpectedValues(3)+(1+Prob\*ExpectedValuesMatrix2(3,k)+(1-Prob)\*ExpectedValuesMatrix1(3,k))\*int2;

 stillcheck=0;

 end;

 k=k+1;

 end;

 if IndicatorAllZeros(PMScheme(j+1:i))==1

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2)); % probability of failure and defect arrival time

 if j==1

 L1=0;

 U1=JobsList(1,2);

 else

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 end;

 int2 =integral2(F,L1,U1,@(x)L2-x,@(x)U2-x);

 F4 = @(x,y)(x+y-sum(JobsList(1:i-1,2))).\*(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2));

 ExpectedTime=integral2(F4,L1,U1,@(x)L2-x,@(x)U2-x);

 ExpectedValues(1)=ExpectedValues(1)+(df+ExpectedValuesMatrix1(1,i)+sum(JobsList(1:i-1,2)))\*int2+ExpectedTime;

 ExpectedValues(2)=ExpectedValues(2)+(1+ExpectedValuesMatrix1(2,i))\*int2;

 ExpectedValues(3)=ExpectedValues(3)+ExpectedValuesMatrix1(3,i)\*int2;

 end;

 end;

 end;

 end;

 end;

 L2=sum(JobsList(1:i,2));

 U2=+inf;

 for j=1:NumberofJobs+1

 if (j>=NumberofJobs)

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2));

 if (j==NumberofJobs+1)

 L1=sum(JobsList(1:NumberofJobs,2));

 U1=+inf;

 else

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 end;

 if U1 ~= inf

 int2 =integral2(F,L1,U1,@(x)L2-x,U2);

 else

 int2 =integral2(F,L1,U1,0,U2);

 end;

 ExpectedValues(1)=ExpectedValues(1)+(sum(JobsList(1:NumberofJobs,2)))\*int2;

 else

 k=j+1;

 stillcheck=1;

 while k <= NumberofJobs && stillcheck==1

 if (j~=NumberofJobs+1) && (j<k) && IndicatorLastoneOthersZero(PMScheme(j+1:k))==1

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2));

 if j==1

 L1=0;

 U1=JobsList(1,2);

 else

 if j <= NumberofJobs

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 else

 L1=sum(JobsList(1:j,2));

 U1=+inf;

 end;

 end;

 int2 =integral2(F,L1,U1,@(x)L2-x,U2);

 Prob=MinimalRepairProb(a,b,U1); % probability of minimal repair

 ExpectedValues(1)=ExpectedValues(1)+(Prob\*(dm+ExpectedValuesMatrix2(1,k))+(1-Prob)\*(dp+ExpectedValuesMatrix1(1,k))+sum(JobsList(1:k-1,2)))\*int2;

 % ExpectedValues(2)=ExpectedValues(2)+ExpectedValuesMatrix2(2,k)\*int2;

 % ExpectedValues(3)=ExpectedValues(3)+(1+ExpectedValuesMatrix2(3,k))\*int2;

 ExpectedValues(2)=ExpectedValues(2)+(Prob\*ExpectedValuesMatrix2(2,k)+(1-Prob)\*ExpectedValuesMatrix1(2,k))\*int2;

 ExpectedValues(3)=ExpectedValues(3)+(1+Prob\*ExpectedValuesMatrix2(3,k)+(1-Prob)\*ExpectedValuesMatrix1(3,k))\*int2;

 stillcheck=0;

 end;

 k=k+1;

 end;

 if (j>=NumberofJobs) || IndicatorAllZeros(PMScheme(j+1:NumberofJobs))==1

 F = @(x,y)(beta1/eta1).\*((x+t0)/eta1).^(beta1-1).\*exp(-1.\*(((x+t0)/eta1).^beta1-(t0/eta1).^beta1)).\*(beta2/eta2).\*((x+y+t0)./eta2).^(beta2-1).\*exp(-1\*(((y+x+t0)./eta2).^beta2-((x+t0)/eta2).^beta2));

 L1=sum(JobsList(1:j-1,2));

 U1=sum(JobsList(1:j,2));

 if U1 ~= inf

 int2 =integral2(F,L1,U1,@(x)L2-x,U2);

 else

 int2 =integral2(F,L1,U1,0,U2);

 end;

 ExpectedValues(1)=ExpectedValues(1)+(sum(JobsList(1:NumberofJobs,2)))\*int2;

 end;

 end;

 end;

 ExpectedValues(1)=ExpectedValues(1)/(1-PZ\_less\_than\_p);

 ExpectedValues(2)=ExpectedValues(2)/(1-PZ\_less\_than\_p);

 ExpectedValues(3)=ExpectedValues(3)/(1-PZ\_less\_than\_p);

end

 function MinimalProb=MinimalRepairProb(a,b,age) % function of probability of either minimal repair or replacement

 MinimalProb=a\*exp(-(b\*age));

 end