Simulation

Task 3

**Bearing Machine**

Group Members:

Ahmad abd El-Latif Mhana(1)

Ahmad Magdy (1)

Ahmad Alaa Abd El-Wahab(1)

Problem formulation

&

Setting the Objective

Problem is that we want to decide whether the current method is better or the proposed one, in money wise.

The current method changes each bearing only if it fails.

The Proposed Method Changes the Three Bearings in the system as soon as any one fails.

Our goal is to decide the better method in money wise.

Model Conceptualization

The system consists of an abstract class “Module”, and class “CurrentModule” & “ProposedModule” inherits from it, I use the current Module to create the proposed module’s data.

This is for the single trial, so for the multiple trials I use “simulationMananger” Class that controls the number of trials I need.

I hold the data in two nested lists, or a table of a class names “bearingItem”, it represents a row in from the current module Bearing life column.

Pseudo code:

* First of all, taking the user input: Downtime cost/Min($), Repair Onsite Cost/Hour ($), Bearing Cost ($), Time One Bearing Change (Min), Time Two Bearings Change(Min), Time Three Bearing Change (Min),Simulation Hours, Simulation Runs, Delay Time Distribution, Bearing Life Distribution
* Then we should create a Simulation for the Current and Proposed Model, for the Simulation Runs, each until it reaches the Simulation Runs.
* For the Current Model: Create a 3 Separate Lives one for each Bearing, each until it reaches the Simulation hours.
* For the Proposed Method, take the resultant Current Model in the Same simulation run, and Loop each Bearing Change, get the minimum life of the 3(if no value create one), until the proposed reaches the Simulation hours limit.
* For(I = 1 : SimulationRuns)

CurrentModelList.Add(CurrentModel curr =New CurrentModel());

ProposedModelList.Add(ProposedModel pro = New ProposedModel(curr));

* Current Model::

For(I = 1 : 3)

Hours = 0;

While(Hours < SimulationHours)

Bearings[i].Add(New Life(), GetDelay())

Hours += Bearing[i].Life;

* Proposed Model::

Hours = 0;

While(Hours < SimulationHours)

Min = MaxValue;

Foreach(BearingItem in CurrentBearingRow)

{

Min = min(Min,(BearingItem == null)? CreateLife() : BearingItem.Life)

}

Hours += Min;

ProposedList.Add(New Proposed(Min,GetDelay());

EXPERIMENTAL DESIGN

For Static Input:

Cost DownTime = 10$

Cost Onsite/Hour = 30$

Cost Bearing = 32$

Time 1 Bearing Change = 20 Min

Time 2 Bearing Change = 30 Min

Time 3 Bearing Change = 40 Min

The length of the simulation runs = 100

The impact of different distributions = Unknown

The impact of the different stopping conditions

For a 20,000 Simulation Hours The Result was:

|  |  |
| --- | --- |
| Current Module | Proposed Module |
| Bearings Cost Avg: 1478.72$  Delay Cost Avg: 3497.5$  Downtime cost avg: 9242$  Pepairpersons avg: 462.1$  Total Cost avg: 14680.32$ | Bearings Cost Avg: 1719.36$  Delay Cost Avg: 1356$  Downtime cost avg: 7164$  Pepairpersons avg: 358,2$  Total Cost avg: 10597.56$ |

For a 10,000 Simulation Hours The Result was:

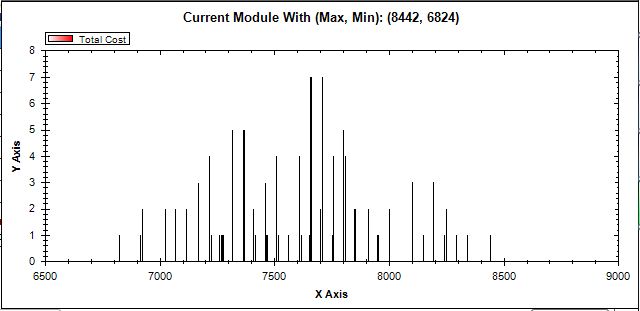
|  |  |
| --- | --- |
| Current Module | Proposed Module |
| Bearings Cost Avg: 765,76$  Delay Cost Avg: 1808.5$  Downtime cost avg: 4786$  Pepairpersons avg: 239.3$  Total Cost avg: 7599.56$ | Bearings Cost Avg: 885.12$  Delay Cost Avg: 705$  Downtime cost avg: 3688$  Pepairpersons avg: 184.4$  Total Cost avg: 5462.52$ |

Results analysis

& Conclusion

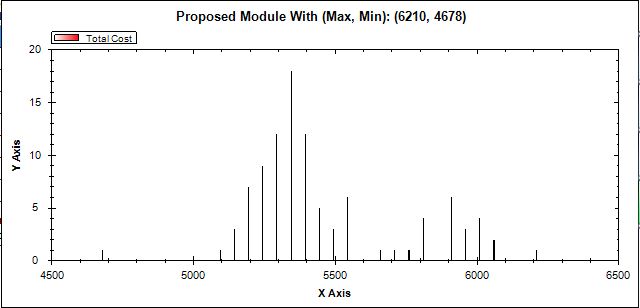
For the Current Method (10,000) simulation Hours

(Max,Min) :: (8442,6824)



For the Proposed Method (10,000) simulation Hours

(Max,Min) :: (6210,4678)



Form

Current(20,000)



Proposed(20,000)



Conclusion

Form the data we have for the current and the proposed method for a 100 trails, we can say the proposed method is better, because it saves thousands of dollars for Any Number of hours of work.