Multiobjective Simulation Optimisation in Software Project Management

D Rodríguez, M Ruiz, JC Riquelme, R Harrison

Outline

- System Dynamics
  - Simulation Opimisation
- Multiobjective metaheuristics
  - NSGA-II
- Multiobjective Simulation Optimisation in Software Project Management

Objective

- To describe an approach that consists of using multiobjective optimisation techniques via simulation (simulation optimisation) to help software project managers find the best values for

  - initial team size and schedule estimates

  for a given project so that

  - cost, time and productivity are optimised.

System Dynamics (SD)

- SD is an approach to understanding the behaviour of complex systems over time.
- SD have being used for modelling software processes development as they provide a framework for analysing the interactions between project activities such as development, testing, etc. and project goals such as deadlines, budget, etc.
- SD can be used to run “what if” simulations for understanding different management policies.
  - Having knowledge of the technical factors of the software processes and the management policies would apply coupled with simulations tools facilitate organizations to improve their processes

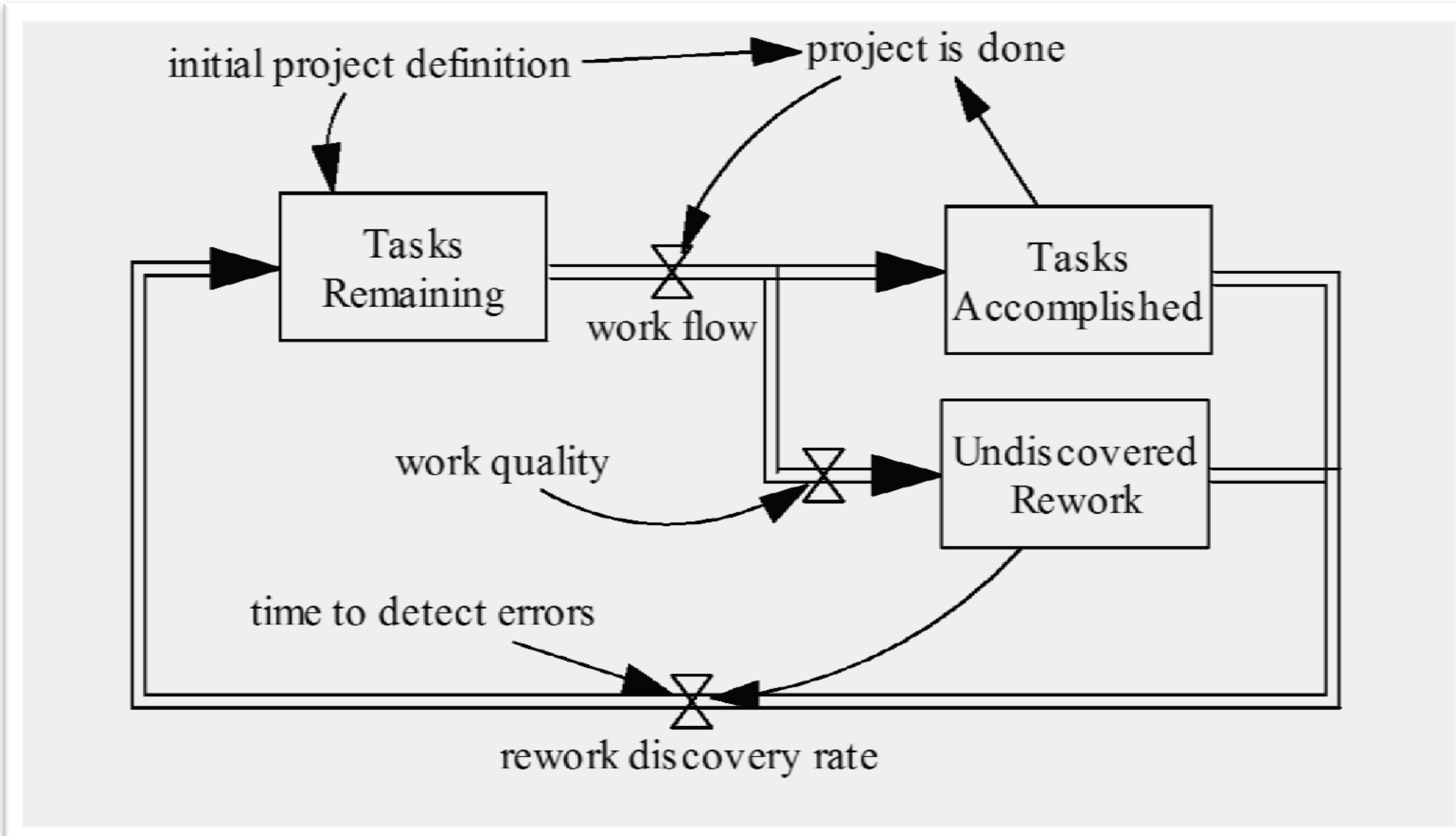
SD Approaches

- Continuous: Based on the analogy of a constant stream of fluid passing through a pipe. The volume may increase or decrease at each time step (fixed), but the flow is continuous.

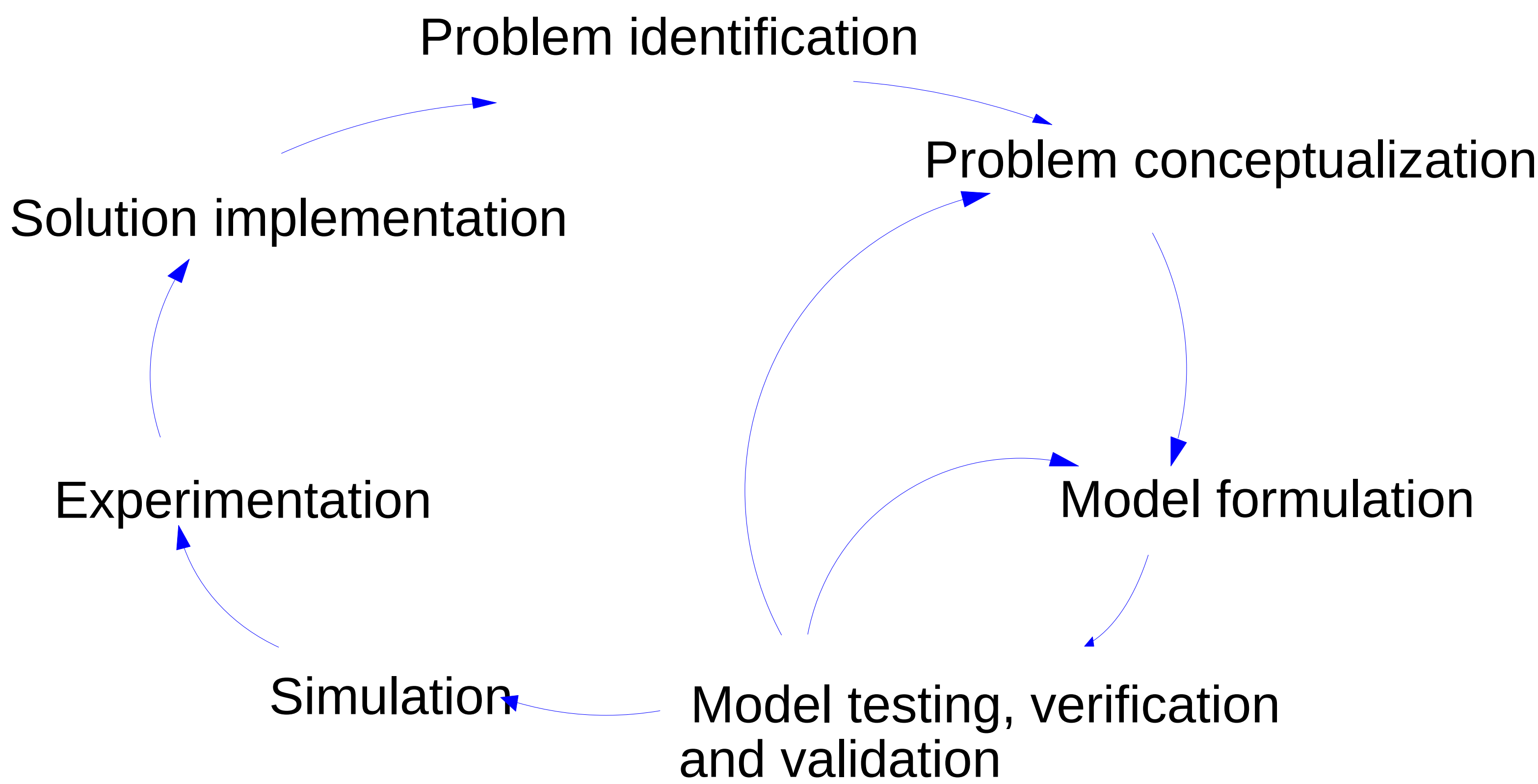
- Discrete: The system changes state as events occur and only when those events occur; the mere passing of time has no direct effect on the model.

- Hybrid simulation: Combine aspects of continuous and discrete event modelling.

SD Example – Continuous



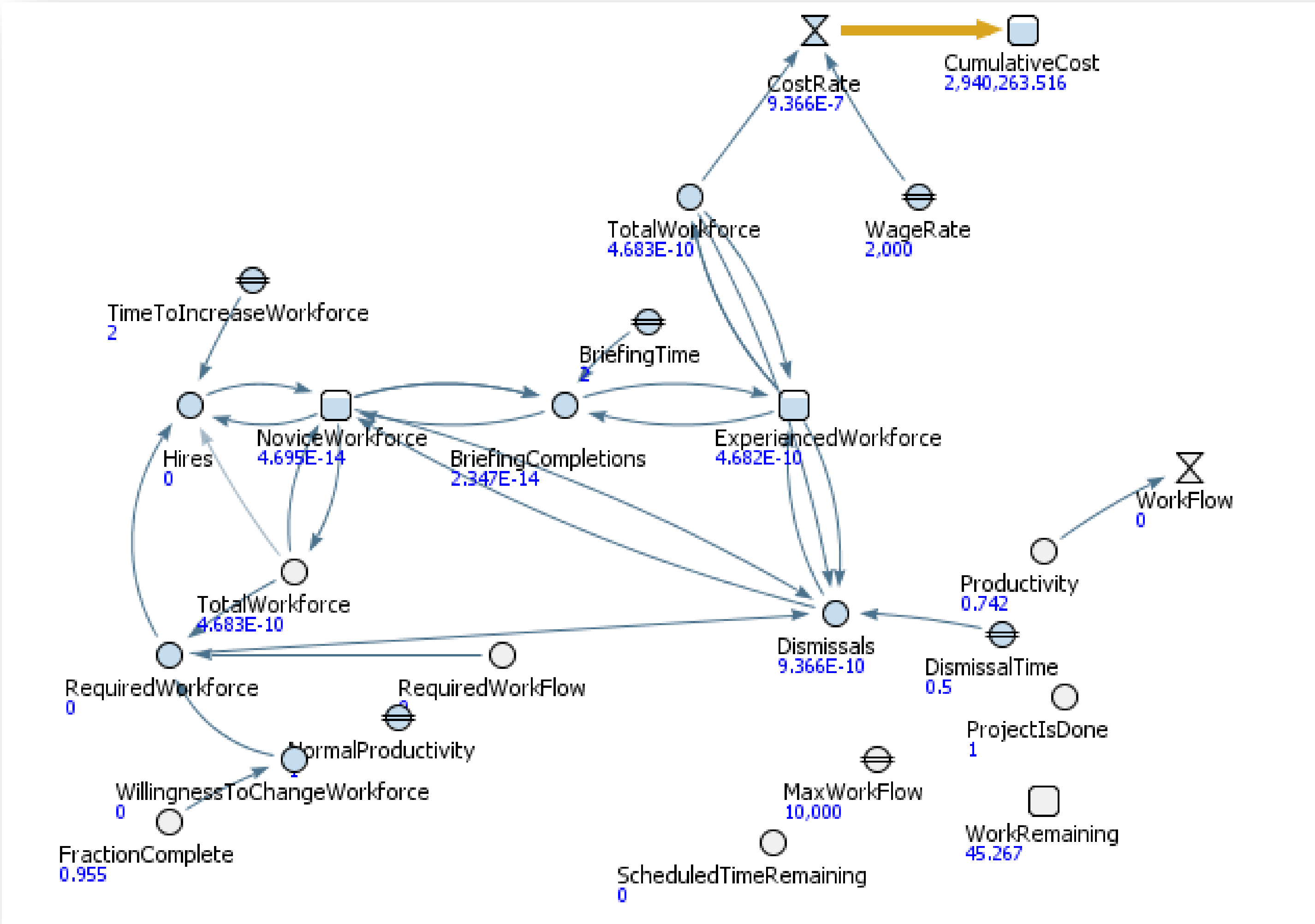
How to Build a Simulation Model?



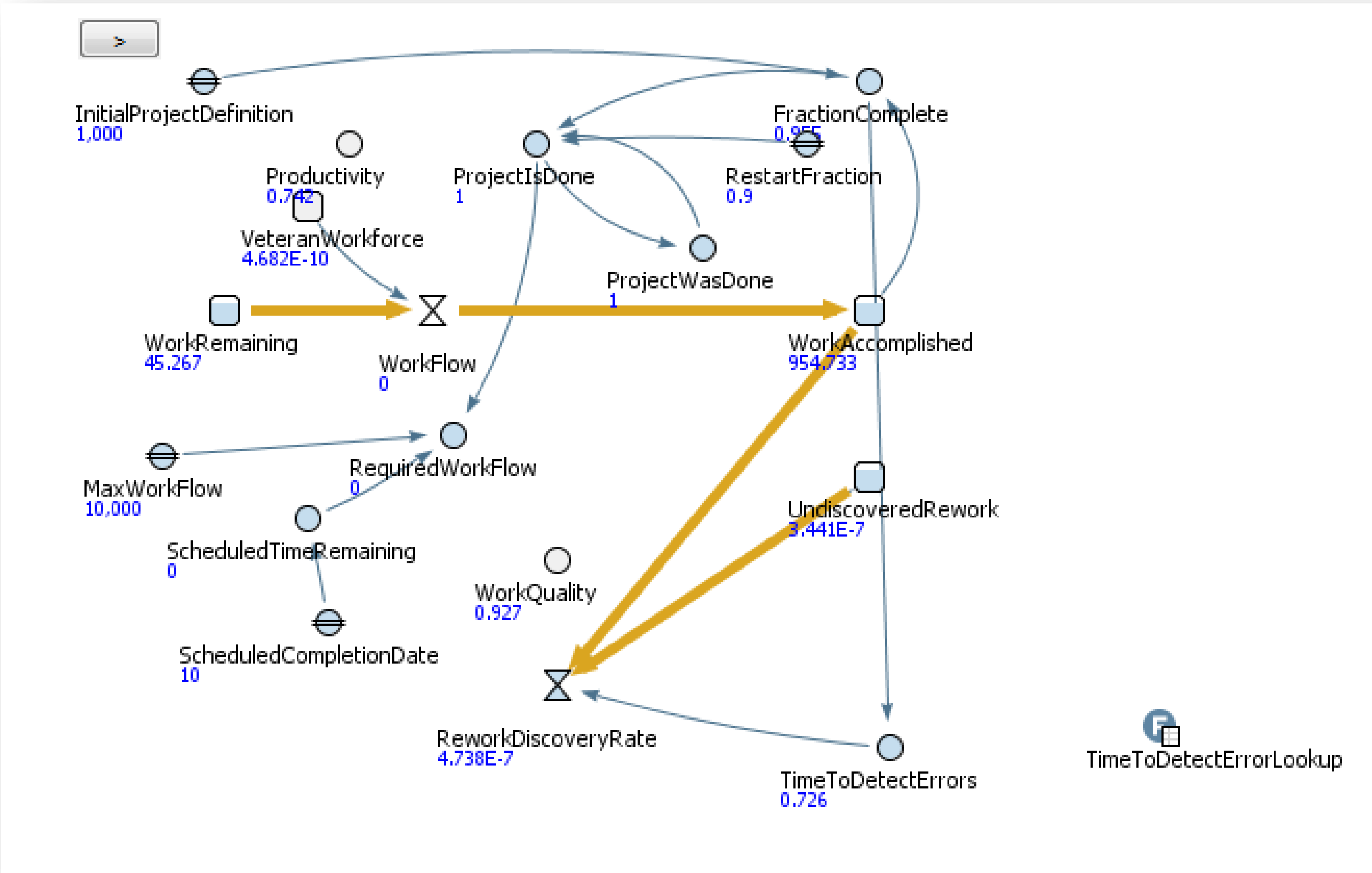
Software Project Management Model

- Composed of 77 feedback loops and 89 equations
- Process abstraction. Structured in three main subsystems:
  - **Development**: This subsystem models the software development process excluding requirements, operation and maintenance
  - **Team management**: It deals with hiring, training, assimilation and transfer of the human resources. It includes Brooks' Law to model training and communication overhead due to team size
  - **Control and planning**: This subsystem gets the initial project estimates and models how and under what circumstances they will be revised through the software project life cycle

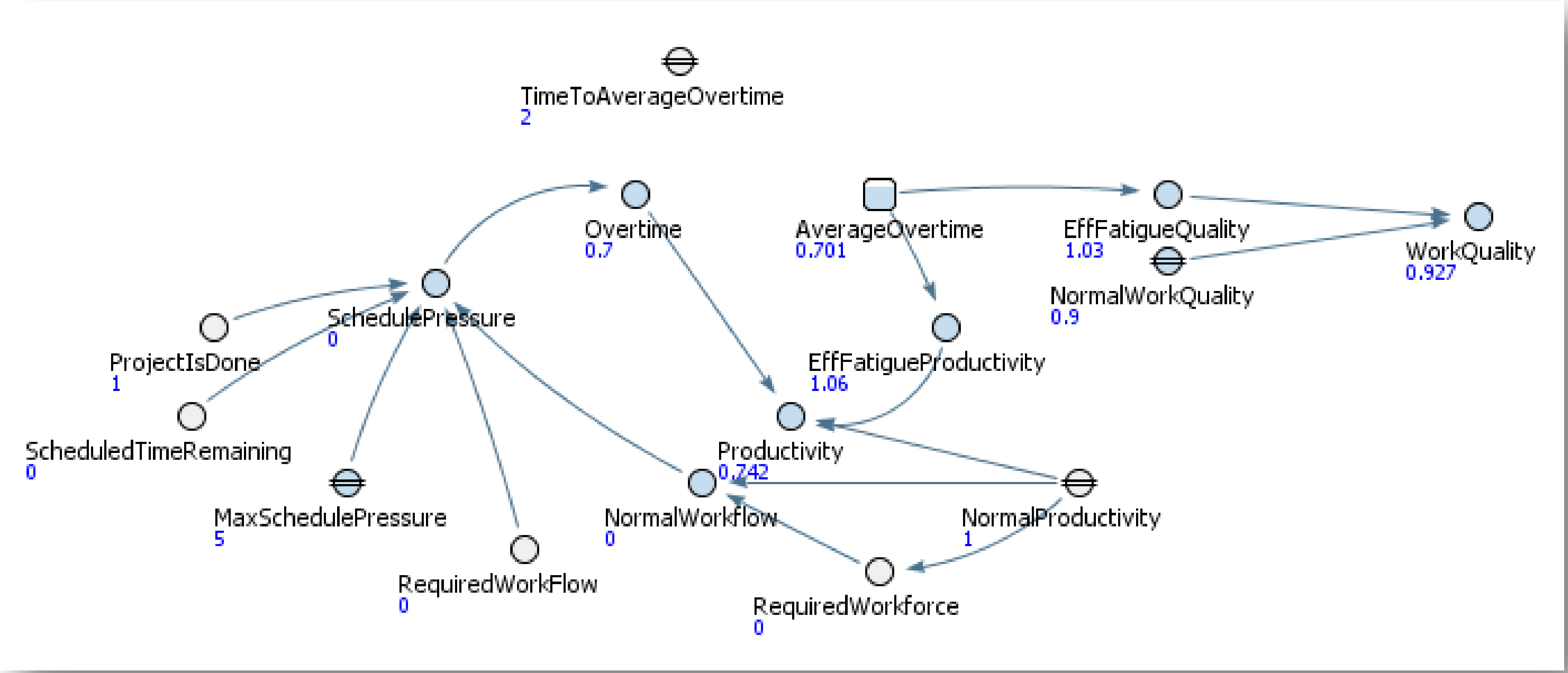
Team Management Subsystem



Development Subsystem



Control and Planning Subsystem

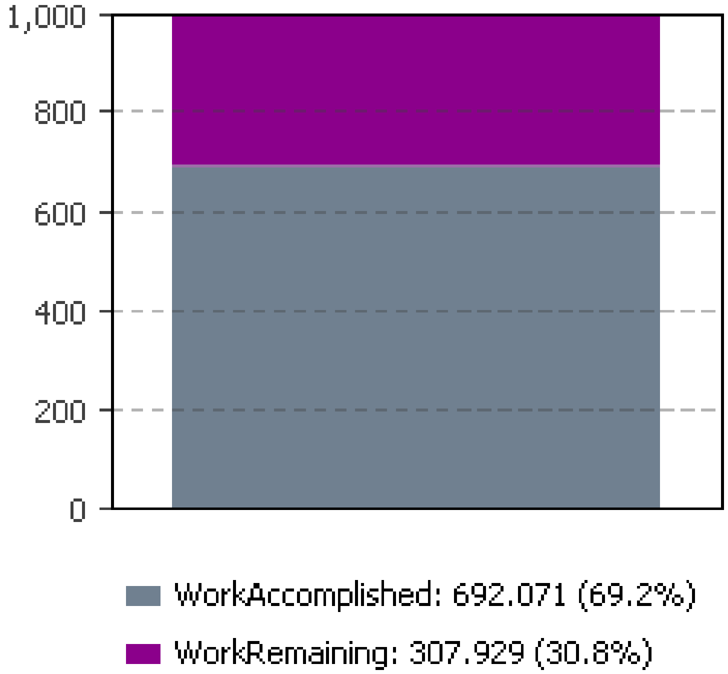


Important Output Variables

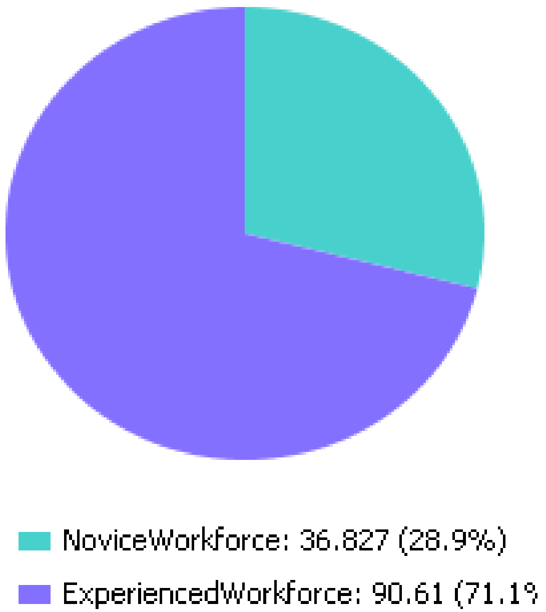
- Output variables
  - *Time*: The final time of the project.
  - *Cost*: The final cost of the project (cumulative cost).
  - *Productivity* : The average productivity reached by the team through the project lifecycle.
- Other output variables that are helpful for analysis during the simulation timeframe are:
  - *Fraction Complete*: The percentage of project completion at any time of the simulation.
  - *Effective Workforce*: The effective work rate performed by the team.

Output variables

Work accomplished
and remaining



Workforce distribution



Important Input Variables

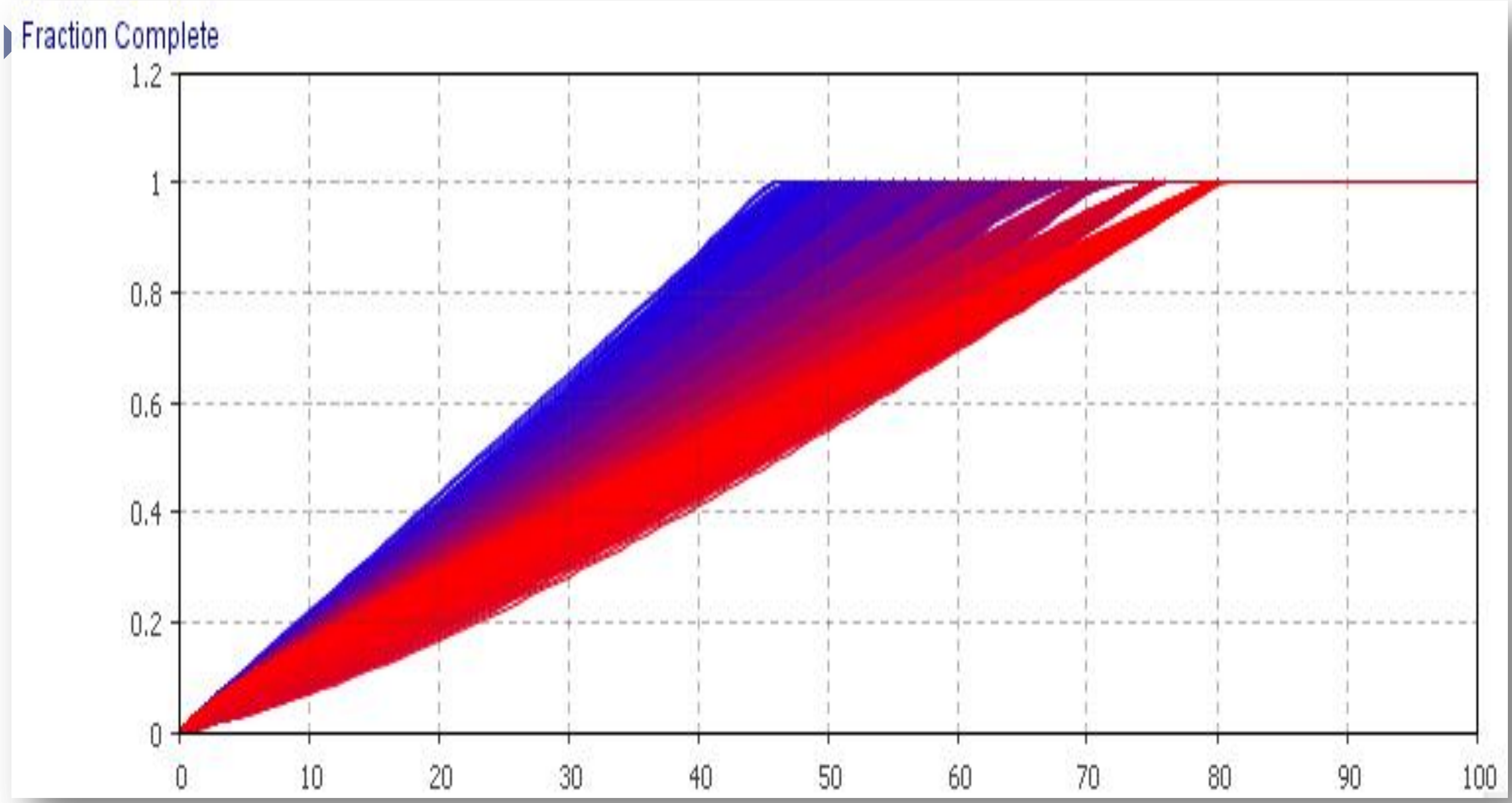
- Initial Novice Workforce (*NoviceWf*):
  - The initial number of novice personnel allocated to the project.
- Initial Experienced Workforce (*ExpWf*):
  - The initial number of experienced personnel allocated to the project.
- Project Size (*Size*):
  - The estimate of project size (we considered Function Points FP as a measure of the size).
- Scheduled Time (*SchldTime*):
  - The estimate of project schedule.

Sensitivity Analysis

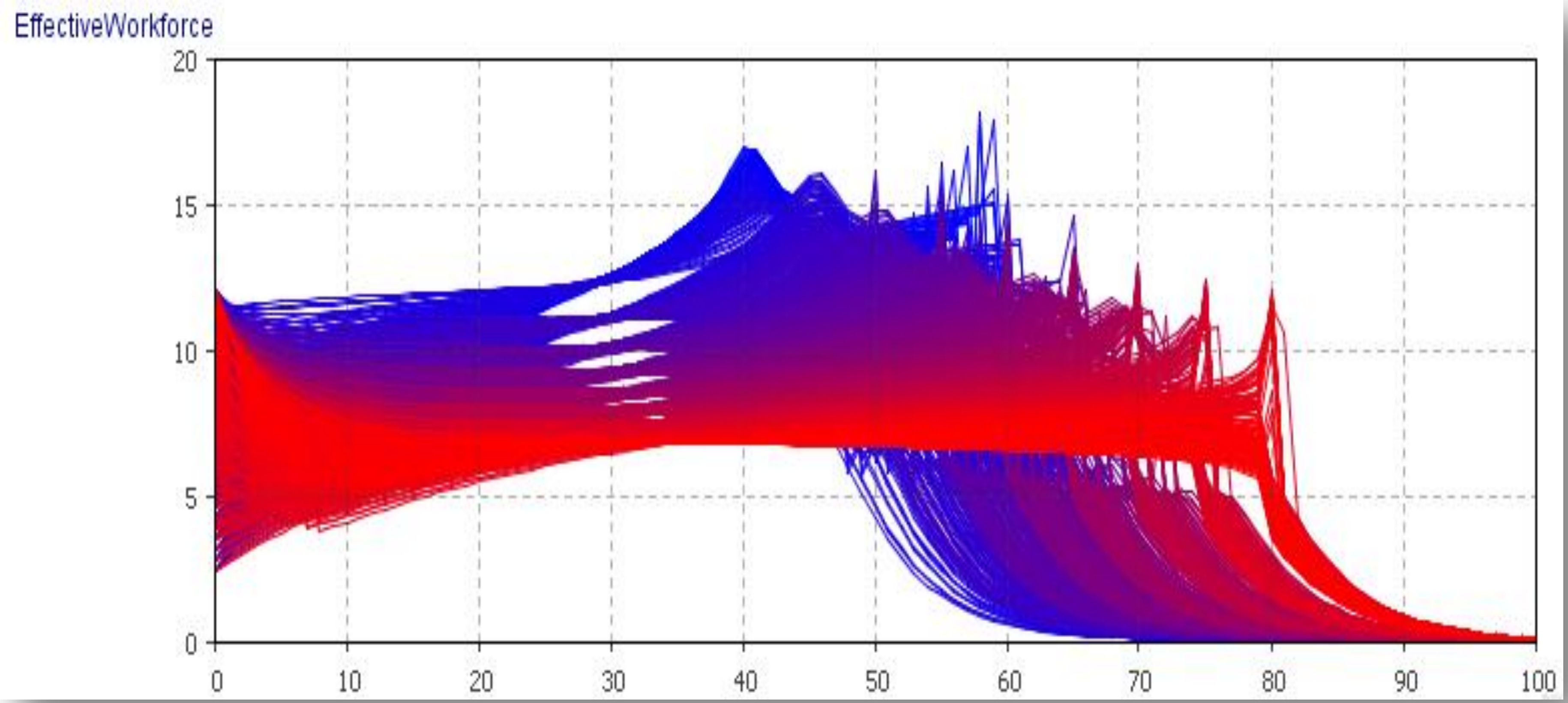
- Using the software project model described, the sensitivity of the output variables regarding productivity, cost and schedule using different initial team size and schedule estimations is determined.
- We designed a scenario for simulation and analysis of the sensitivity of the main output variables to the variation of the main input parameters.

Inputparameter	*Range*	*Step*
InitialNewWorkforce	[0-10]	1
InitialVeteranWorkforce	[2-10]	1
ScheduledCompletionTime	[45-80]	5

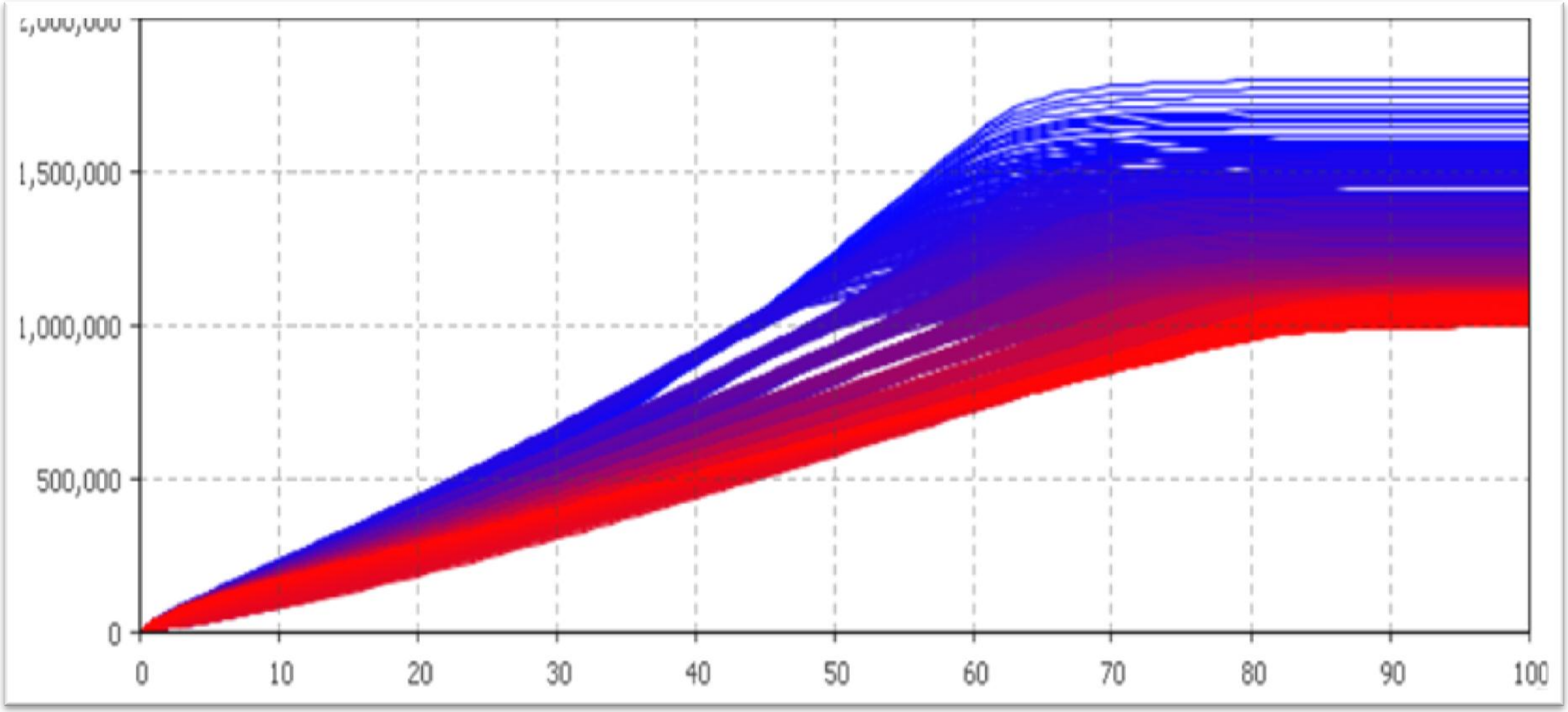
Sensitivity Analysis – *Fraction Complete*



Sensitivity Analysis – *Effective Workforce*



Sensitivity Analysis – *Cumulative Cost*



Simulation Optimisation

- Defined as the process of finding the best values of some decision variables for a system where the performance is evaluated based on the output of a simulation model of this system.

- In our case, once the sensitivity of the output variables of the model has been determined, the next step for the project manager should be to use the model in order to decide what values of the input parameters optimise the key project indicators.

Single Optimisations

- Current simulation tools provide a single fitness function
  - All objectives need to be aggregated to form a single objective or a scalar fitness function which is then treated by some classical techniques, mostly simulated annealing and scatter search.
- This approach brings problems regarding how to normalise, prioritise and weight the different objectives in the global fitness function.
  - In software project management it is also usual that conflicting objectives interact with each other in nonlinear ways.
- Finding an adequate function becomes critical in this approach since the set of solutions is highly dependent upon the function selected and the weights assigned.

Single Optimisation – Results

- It can be observed that the initial team size and the scheduled completion time vary depending on the objective one wants to achieve.
  - Not a very realistic situation in software project management, since project managers would be interested in the combination of input parameters that lead to the project with the maximum productivity and the minimum cost and time.

Output		Input Parameters		
		$NoviceWf$	$ExpWf$	$SchldTime$
$Cost$	\$992K	0	10	80
$SchldTime$	44.75	3	10	40
$Prod$	11.47	3	10	40

Multi-objective Optimisation Problems

- Multi-objective Optimisation problems (MOP) are those that involve multiple and conflicting objective functions.
- MOP is also known as Multiple Criterion Decision Making (MCDM) in other fields such as in operation research.
- In general, the solutions for MOP are defined using the Pareto front.

Deb et al's NSGA-II Algorithm

Algorithm 1 NSGA-II Algorithm [17]

$P_0 \leftarrow$ **makeInitalRandomPopulation**() $\triangleright$ Initial Population of size N
$Q_0 \leftarrow$ **makeNewPopulation**(P_0)
$R_0 = \emptyset \leftarrow \wedge\ t \leftarrow 0$
while $t \leq max\_generations$ **do**
    $R_t \leftarrow P_t \cup Q_t$ $\triangleright$ Combine parent and offspring populations
    $\mathcal{F} \leftarrow$ **fastNonDominatedSort**(R_t)
    $P_{t+1} \leftarrow \emptyset \wedge i \leftarrow 1$
    while $|P_{t+1}| + |\mathcal{F}_i| \leq N$ **do** $\triangleright$ While population size is not full
        crowdingDistance($\mathcal{F}_i$) $\triangleright$ Calculate crowding measure in $\mathcal{F}_i$
        $P_{t+1} \leftarrow P_{t+1} \cup F_i$ $\triangleright$ Include the ith rank into the population
        $i \leftarrow i + 1$
    end while
    Sort($\mathcal{F}_i, \prec_n$) $\triangleright$ Sort in descending order using $\prec_n$
    $_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i[1 : (N - |P_{t+1}|)]$ $\triangleright$ Fill population untill size N
    $Q_{t+1} \leftarrow$ **makeNewPopulation**(P_{t+1})
    $t \leftarrow t + 1$
end while
return $\mathcal{F}_1$ $\triangleright$ Return the best *Pareto rank*

Deb et al's NSGA-II Algorithm

- NSGA II is popular algorithm that uses
  - Crowding distance measures how far away an individual is from the rest of the population
  - Wider Pareto front
  - Elitisim
  - Non-dominated sorting in each generation
  - The most used algorithm!

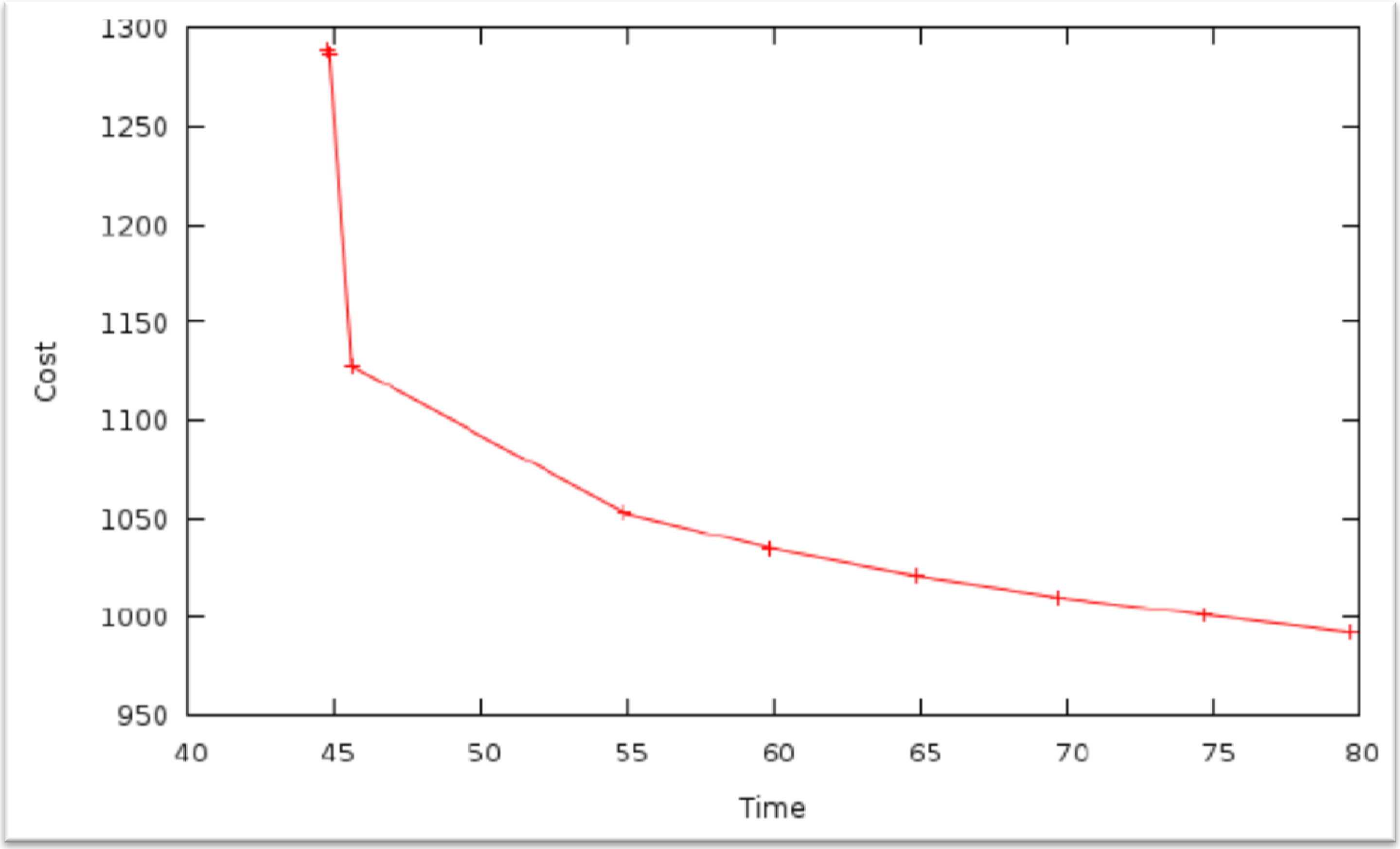
  - JMetal implementation
    - `http://jmetal.sourceforge.net/`
    - Durillo and Nebro (2011)

NSGA II Output for Objectives *Time* and *Cost*

$NoviceWf$	$ExpWf$	$SchldTime$	$Time$	$Cost$ (\$(K))
3	10	40	44.75	1,289
2	10	45	44.85	1,287
1	10	45	45.58	1,128
0	10	55	54.90	1,053
0	10	60	59.84	1,035
0	10	65	64.79	1,021
0	10	70	69.72	1,010
0	10	75	74.71	1,001
0	10	80	79.68	992

Objectives *Time* and *Cost*

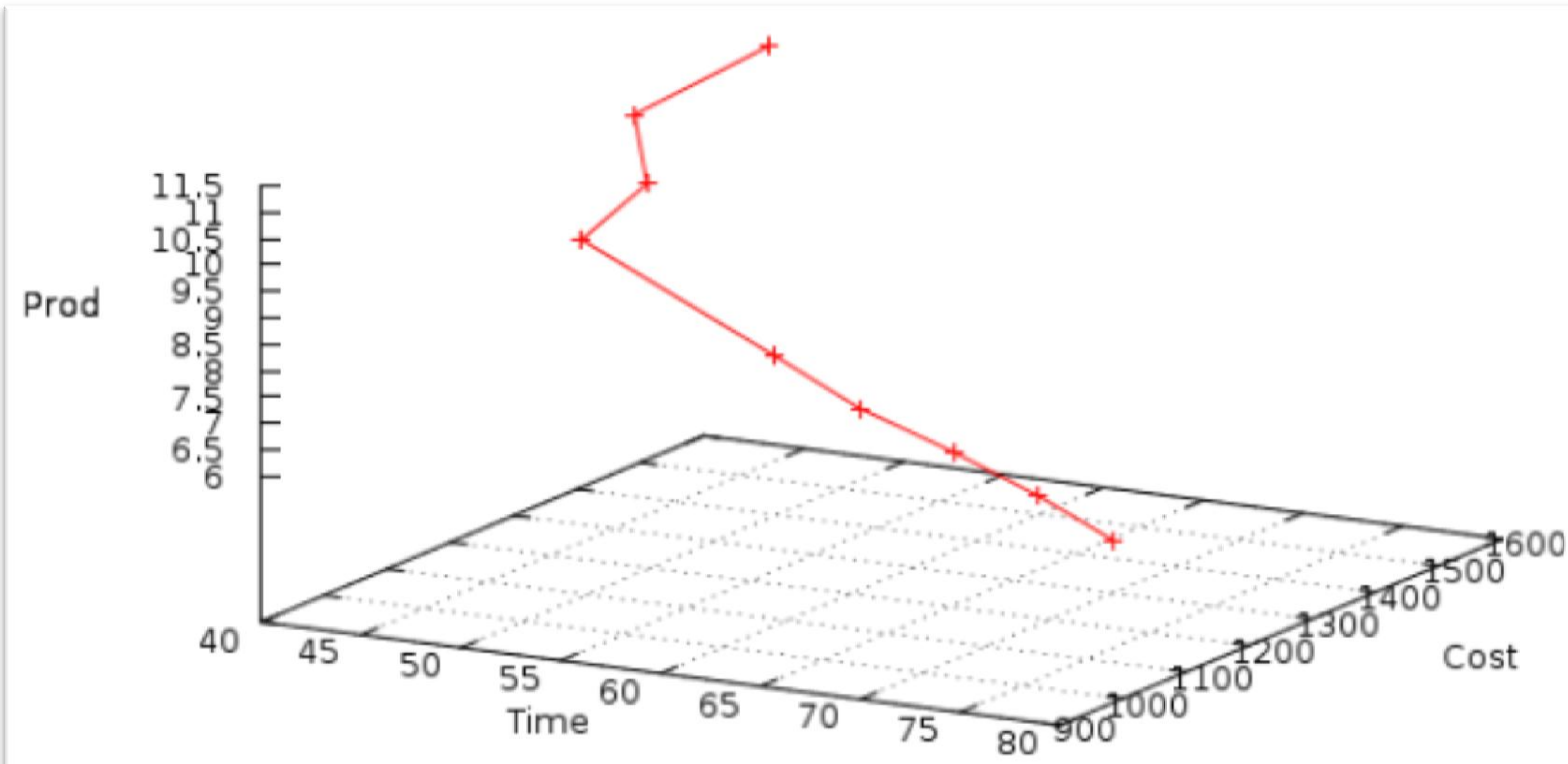
- Graphical output for two objectives, *Time* and *Cost*



NSGA-II Output with Three Objectives

- *Time*, *Cost* and *Productivity*

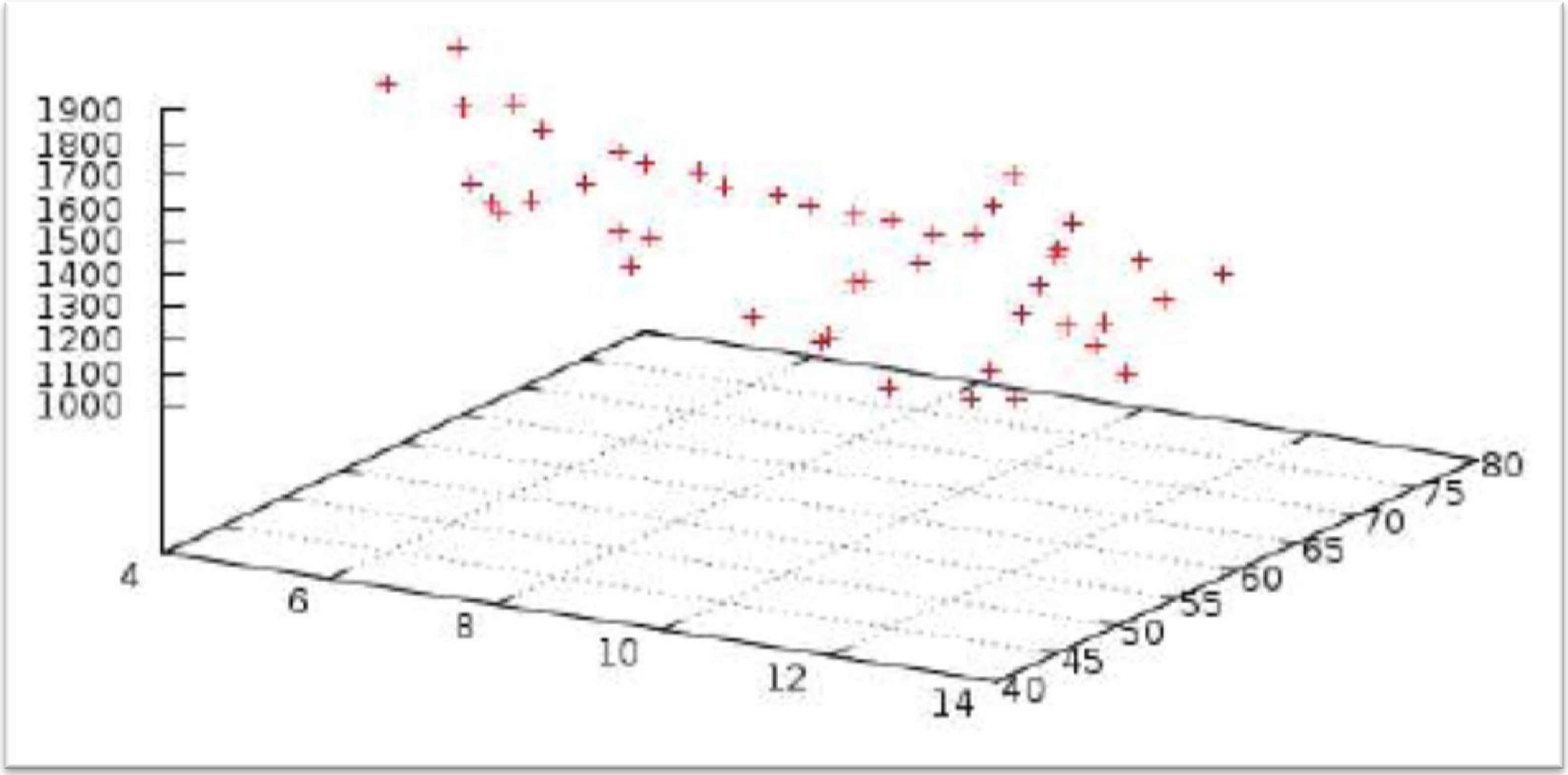
$NoviceWf$	$ExpWf$	$SchldTime$	$Time$	$Cost$ ($(K))	$Prod$
5	10	35	44.96	1,548	11.12
4	10	40	45.47	1,318	10.99
5	10	45	48.72	1,235	10.26
1	10	50	50.00	1,091	9.99
3	8	60	60.00	1,082	8.33
3	9	65	64.90	1,060	7.70
4	10	70	69.85	1,052	7.15
3	10	75	74.78	1,030	6.68
0	8	80	79.69	993	6.27
0	10	80	79.68	992	6.27



Prod
11.5
11
10.5
10
9.5
9
8.5
8
7.5
7
6.5
6
40
45
50
55
60
65
70
75
80
Time
900
1000
1100
1200
1300
1400
1500
1600
Cost

With 5 Objectives…

- Previous objectives, plus the minimisation of both the no. of experienced personnel and the addition of novice and experienced personnel.



Conclusions

- Multiobjective optimisation techniques applied to simulation models give project managers better control over the set of input variables than single optimisations.
- Multobjective techniques can lead to achieve better results in terms of finding the input parameters that will maximise output parameters.
  - No need to calculate the weights when combining the conflicting objectives into a single one.
  - The range of solutions helps with understanding different project policies.

Future Work

- Further models
  - Discrete and hybrid simulation are suitable simulation techniques for Sofware Engineering problems
- Comparison with other multiobjective techniques
  - In particular with SPEA-2
  - PAES for high dimensional data and scalability studies
- Visualisation and clustering techniques of the Pareto fronts

Thanks!