A Spreadsheet-like User Interface for Combinatorial Multi-Objective Optimization

Derek Rayside
MIT CSAIL
[Waterloo 2010]

H.-Christian Eslter
University of Paderborn
[MIT 2008]

Concorde SST

Airbus A380

Boeing 787

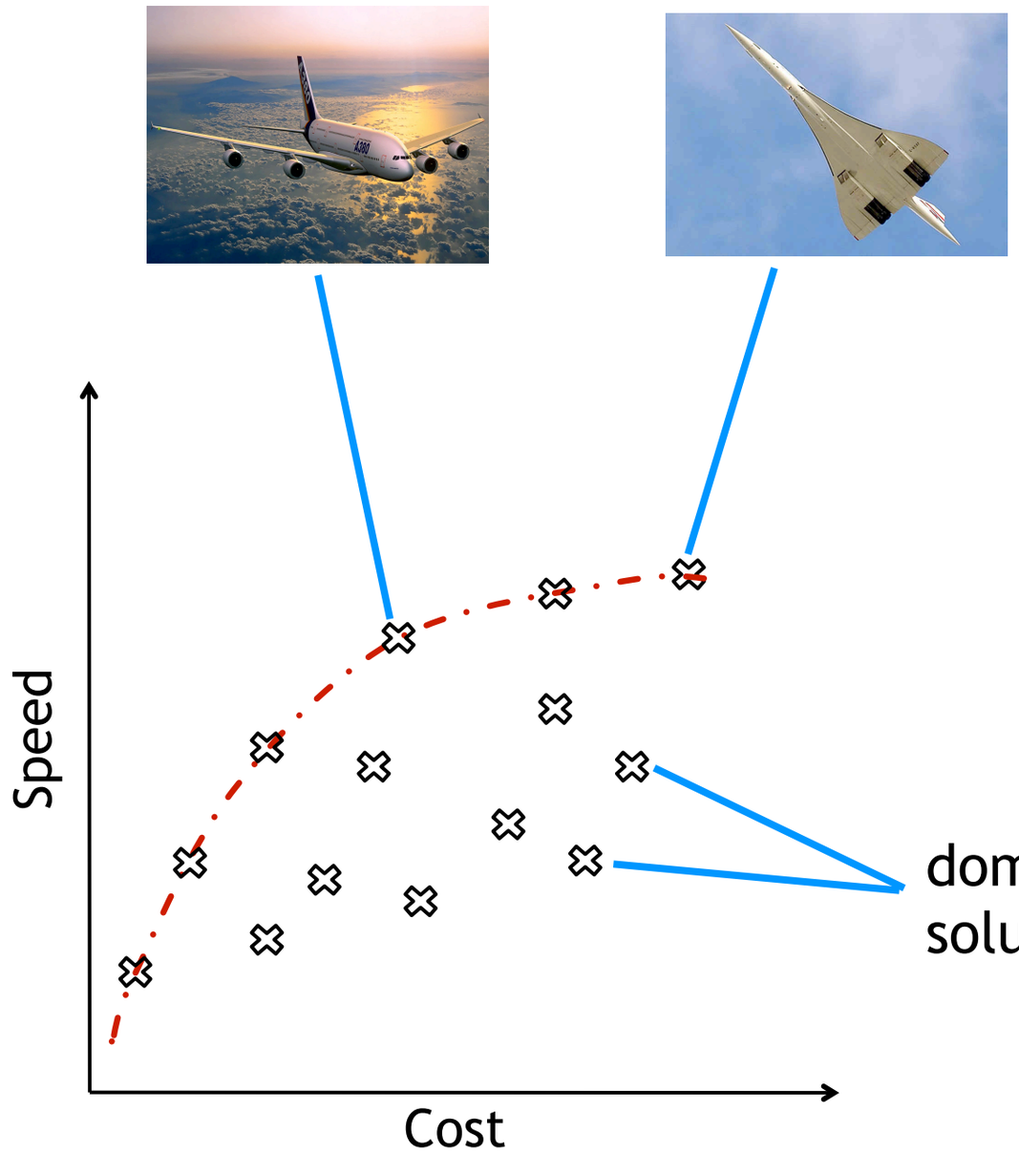
Speed is King

Multiple Objectives

Choose Two

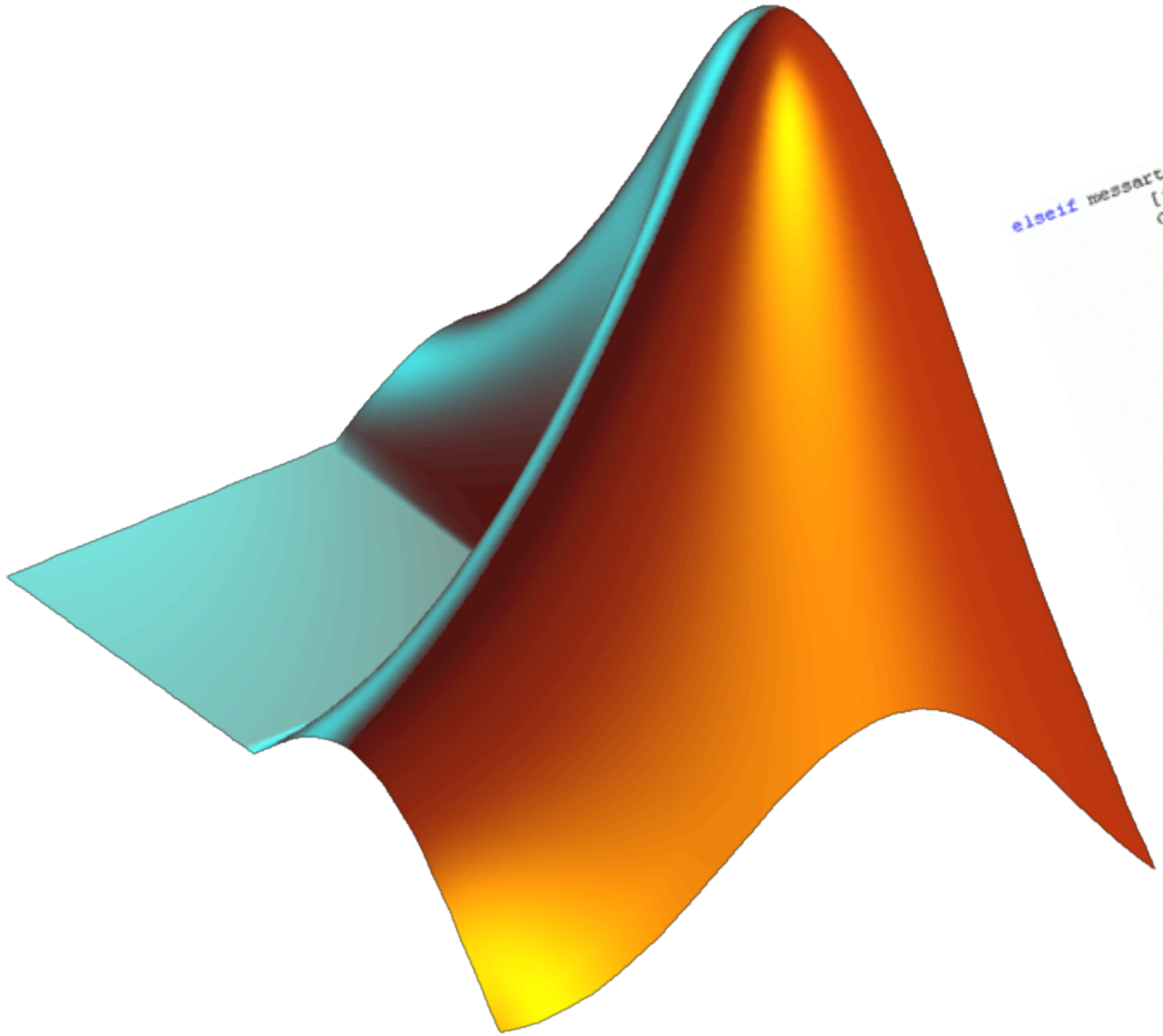
— Arthur C. Clarke

Pareto Front



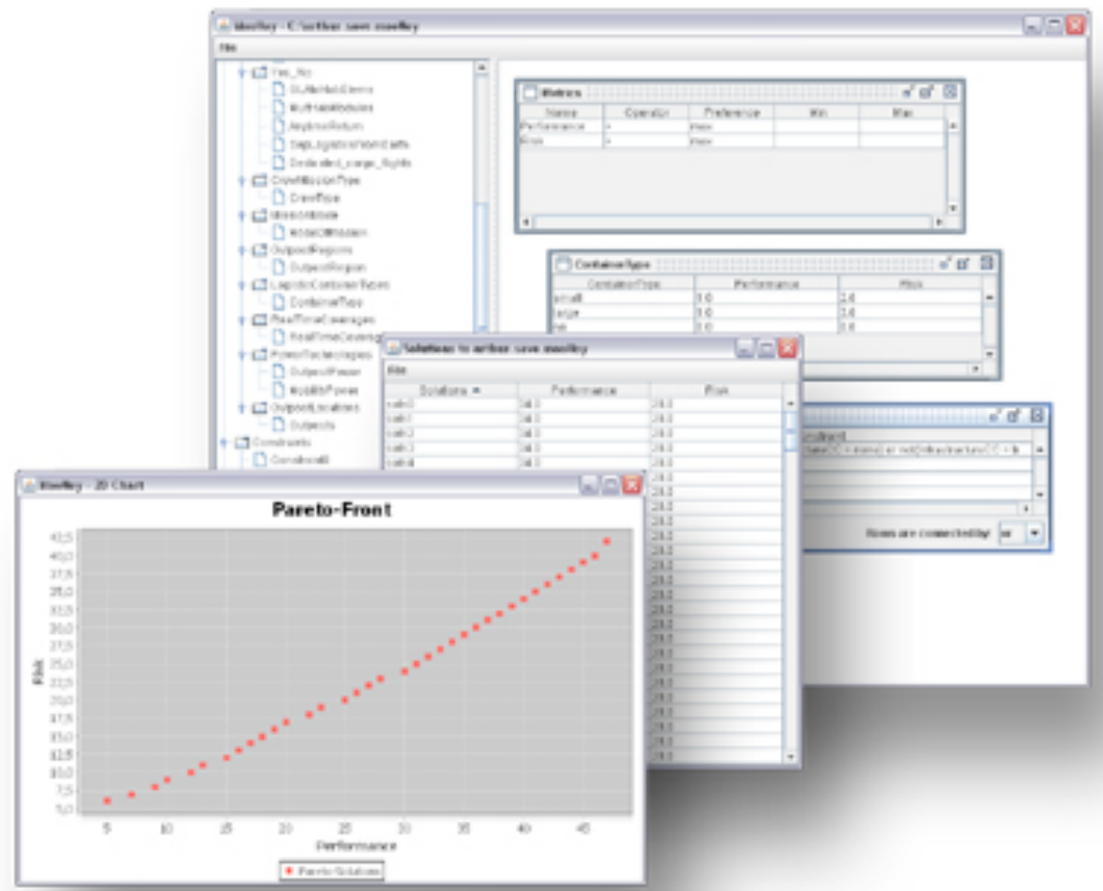
How can I get started?

How can I get started?



```
elseif messart(ilag)==2
    [indz,inds] = find(C1(1:izanf+mlkoord-1,
    C1(indz,NFG+1:2*NFG) = C1(indz,1:NFG);
    C1(indz,1:NFG) = zeros(length(indz),NFG
    if Nas~=0
        switch language
        case 'D'
            fprintf('\n\n FEHLER: Geschw
        case 'E'
            fprintf('\n\n ERROR: Vel. c
        end
        return
    end
    izanf = izanf+mlkoord;
end
end
% Recalculate controller matrix B in
if ANZKOP > 0
    B1 = B1*TC2;
end
    % Transform B1*C1 to indices for
Tilting/Translation Control
    %B1NFG = B1(1:2*NFG,:); % B1 for
    %C1NFG = C1(:,1:2*NFG); % C1 for
    %if Nas ~= 0
    %    B1Nas = B1(2*NFG+1:2*NFG+
    %    C1Nas = C1(:,2*NFG+1:2*NF
    %end
```

Moolloy



Multi-Objective Optimization with Alloy

Outline

- Introduction
- Example: bicycle configuration
- Definition of a Multi-Objective Problem
- The Moolloy User Interface
- User Case Studies
- Conclusions & Future Work

Bicycle Configuration

Metrics: Speed vs Cost

Frame: aluminum or steel?

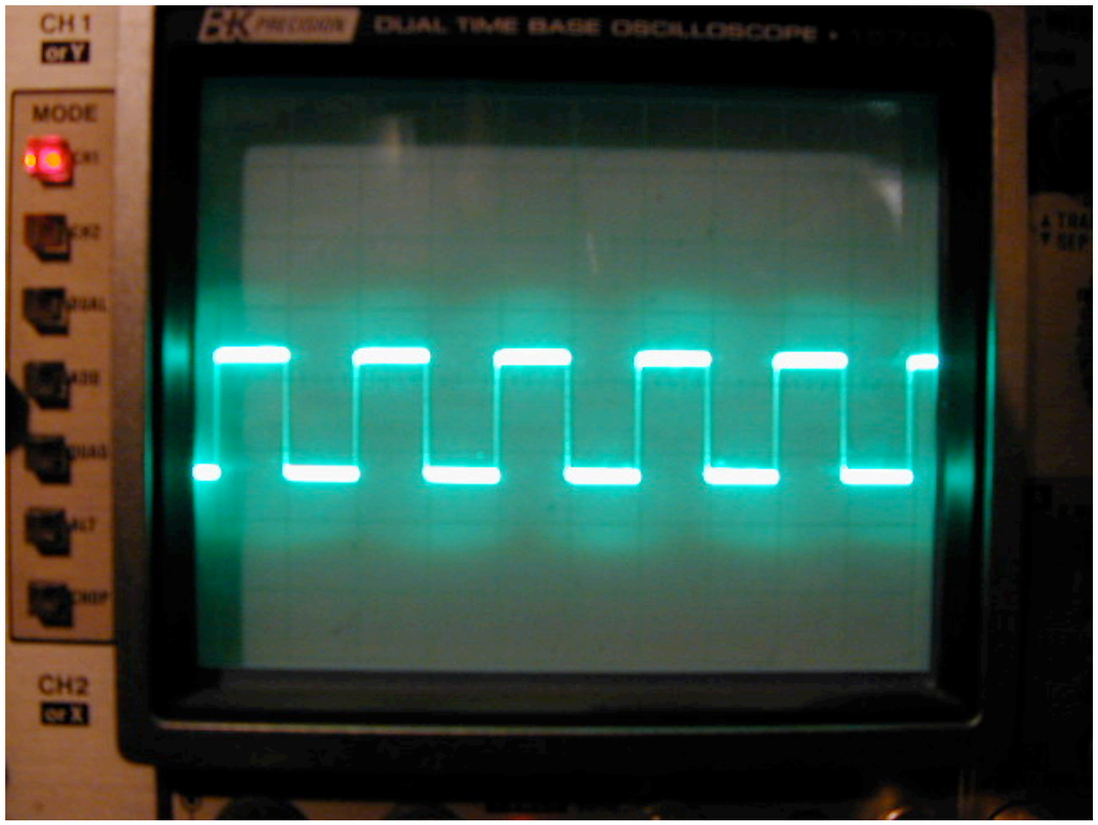
Forks: regular or shocks?

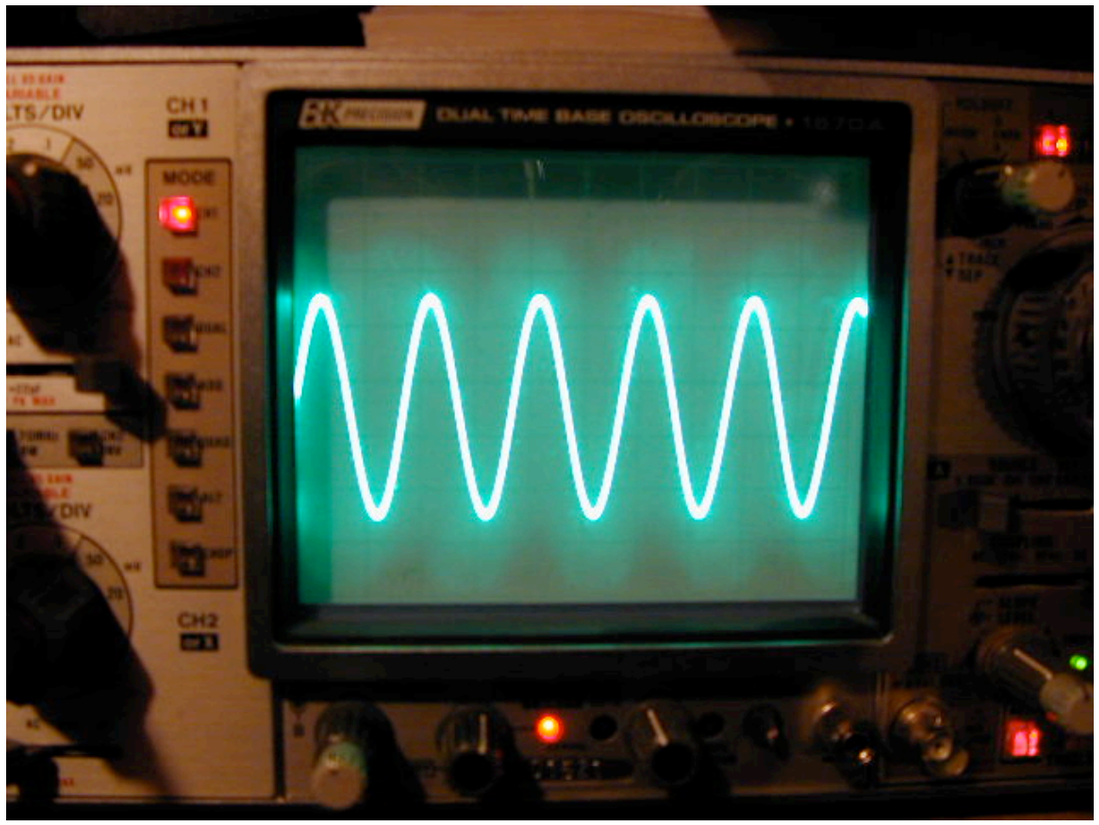
Handlebars: racing, etc, ?

Definition of a MOOP

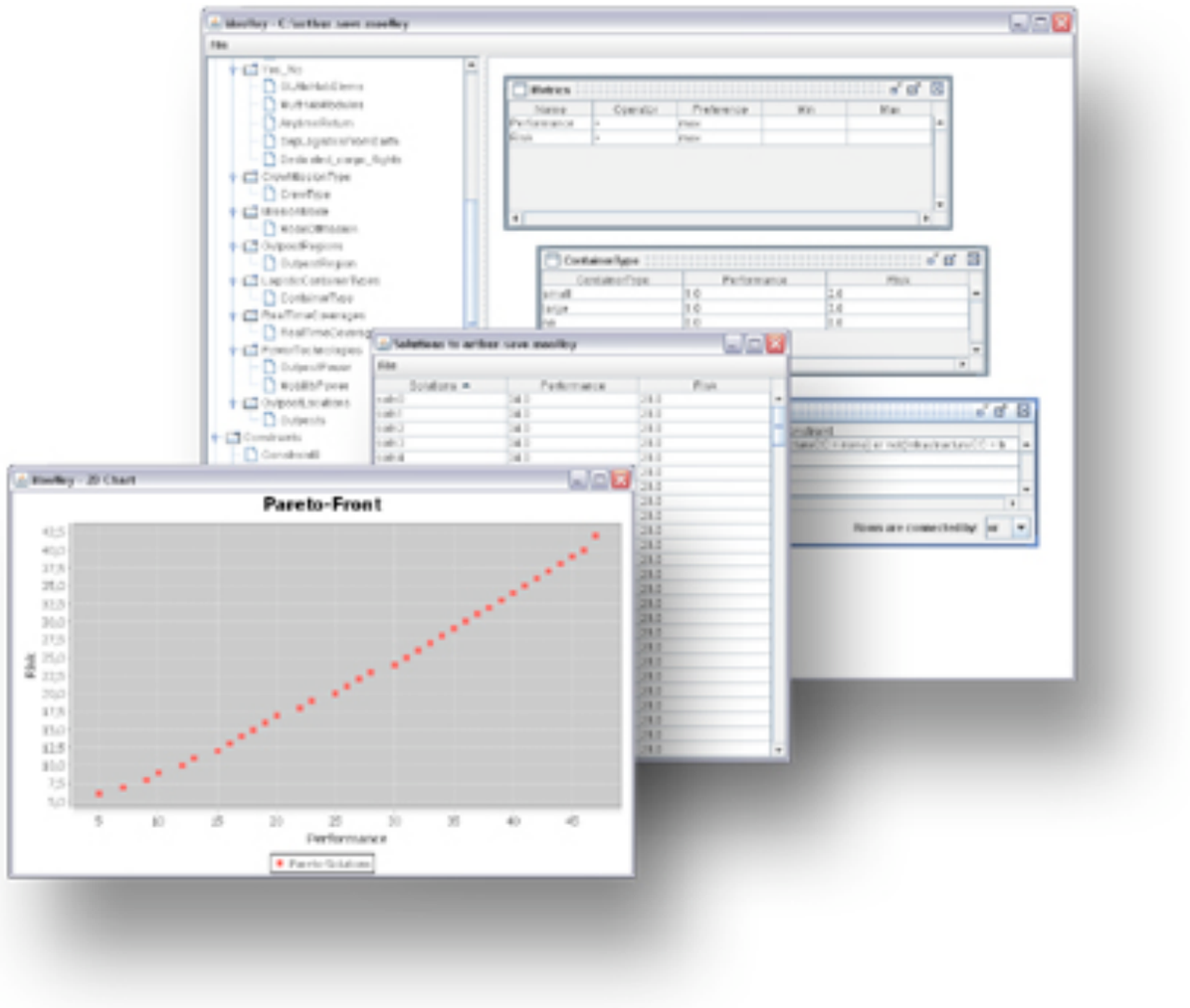
Elements	Examples
Decision variables: $x_1, \ldots, x_z$	Frame; Fork; Front mech; Rear mech;
Domains: $d_1, \ldots, d_y$	{Steel, Aluminum}; {RockShock_XL, LST_AIR}; {Srum9, Srum10, Shimanu_XR};
Constraints: $c_1, \ldots, c_p$	if frame material is aluminum then fork model must be RockShock_XL
Metrics: $m_1, \ldots, m_q$	Cost; Performance;

Discrete Domains

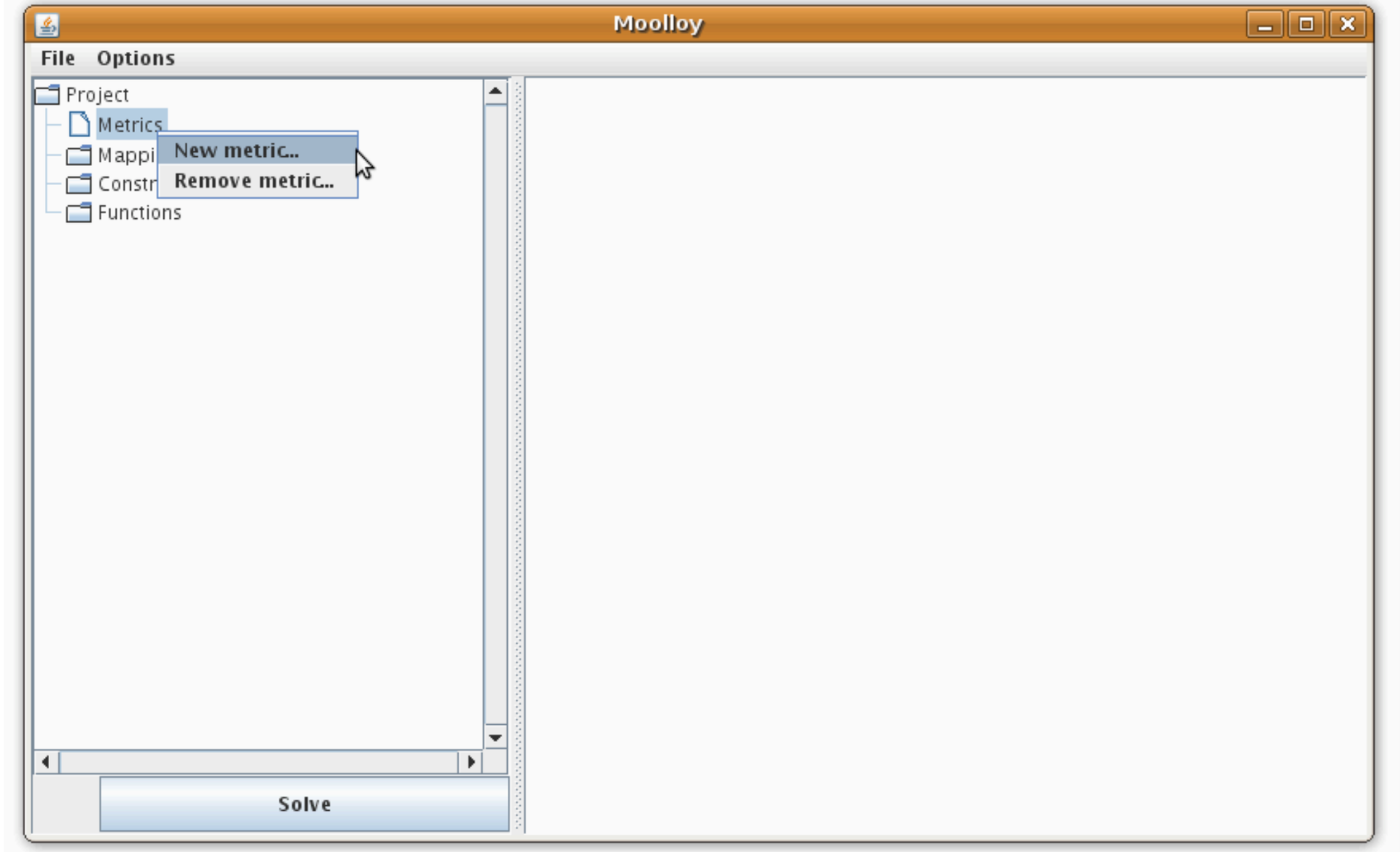




Moolloy User Interface



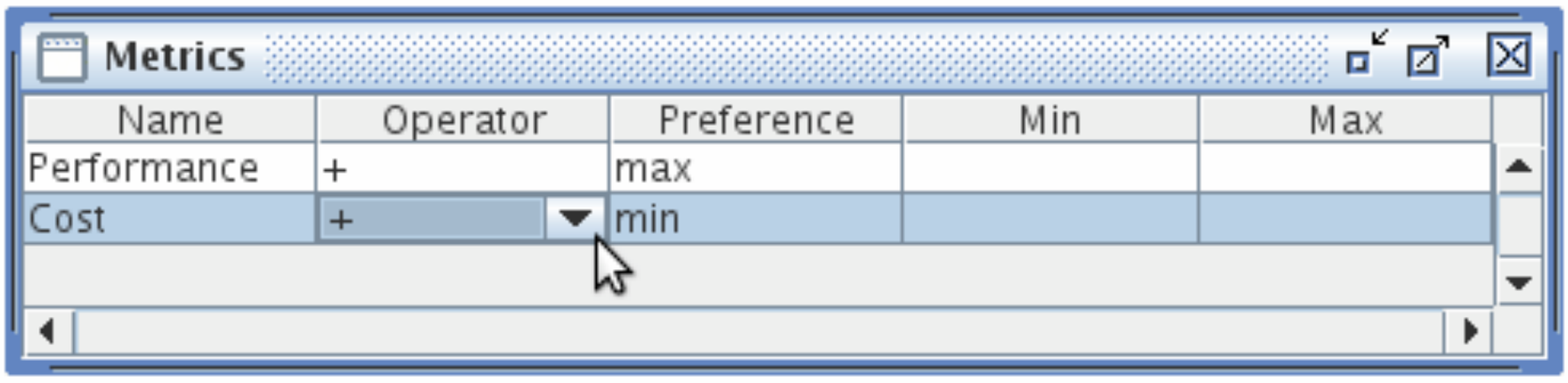
Project Tree



Declaring Metrics

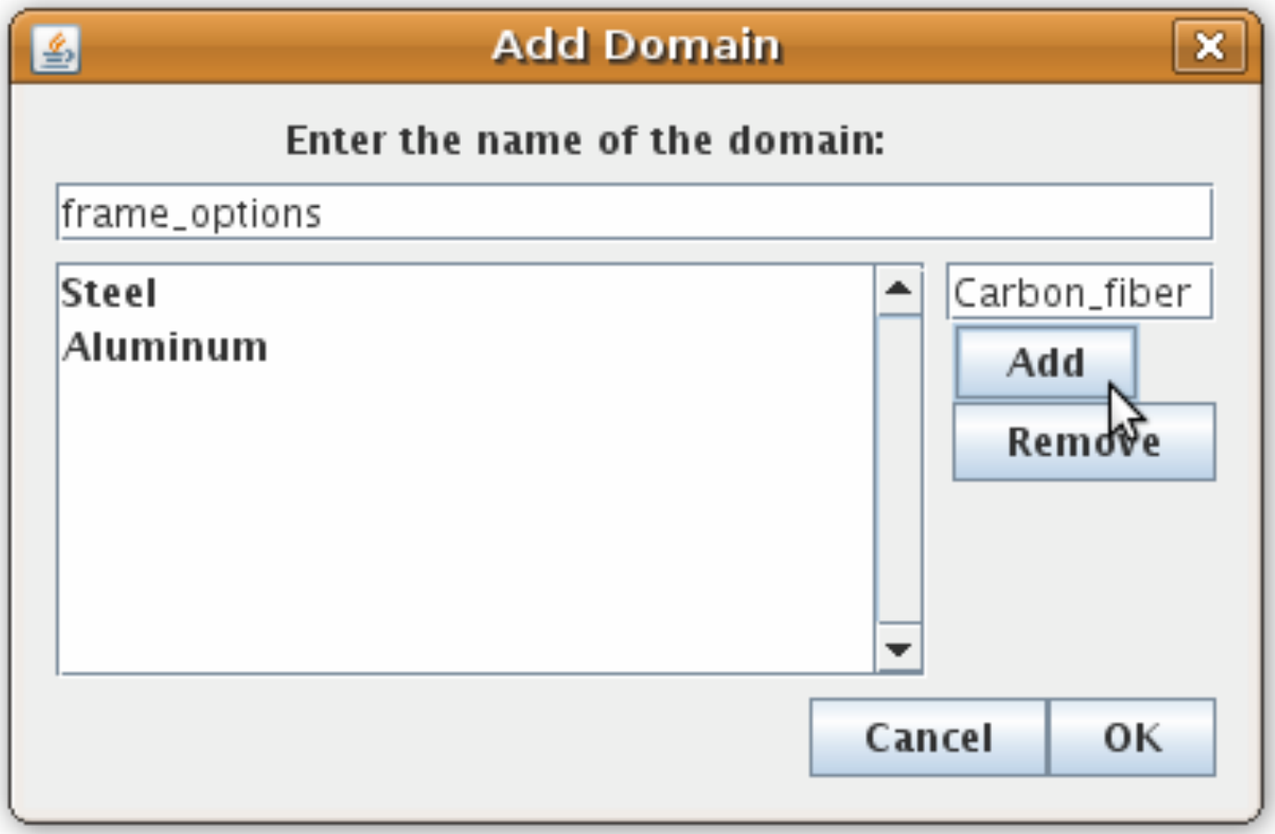
- Start with declaring *Metrics*

- Choose *Operator* : *, +
- Choose *Preference*: min, max, none



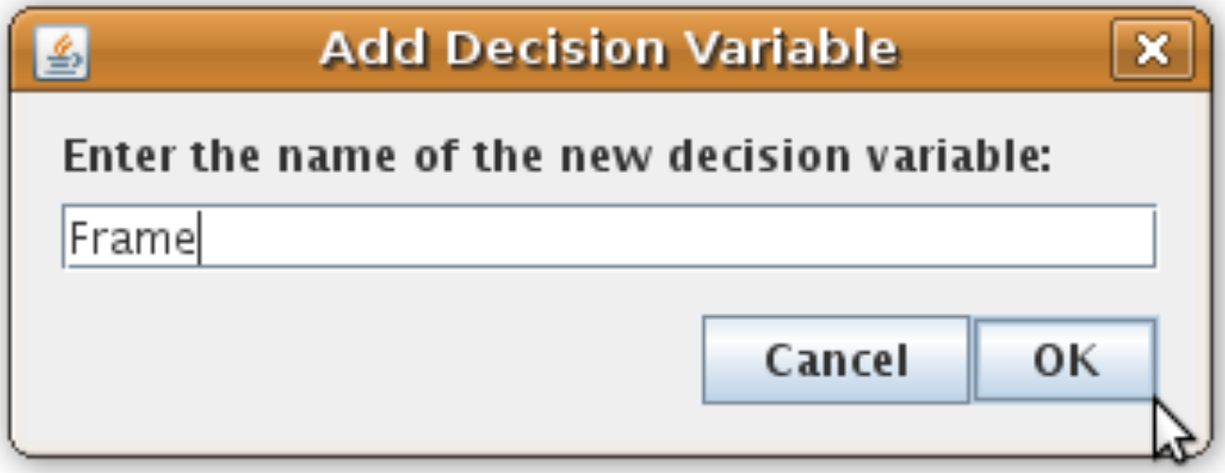
Declaring Domains

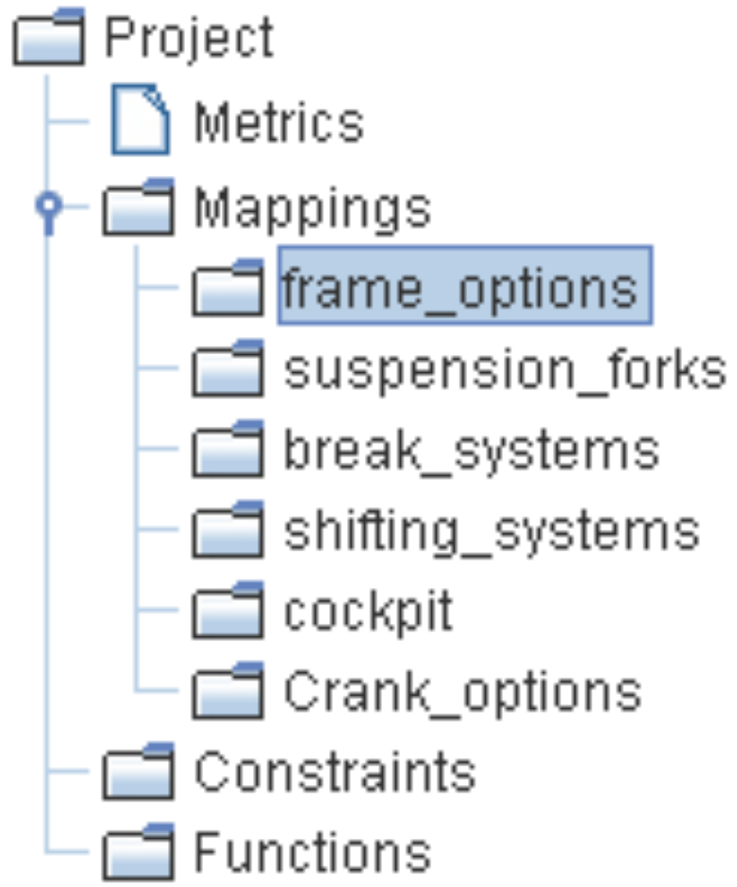
- Add domains and their values
- Examples:
  - *frame_options*: {Steel, Aluminum, Carbon_fiber}
  - *forks*: {regular, shocks}



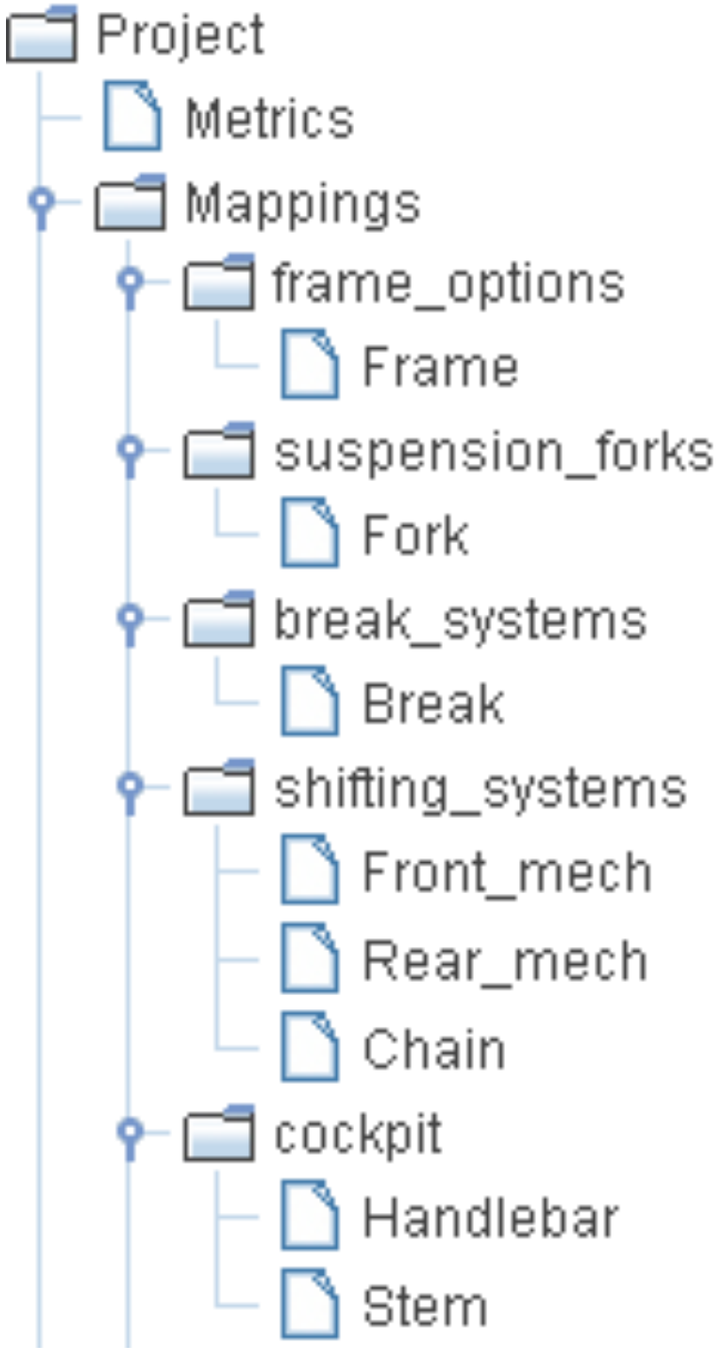
Decision Variables

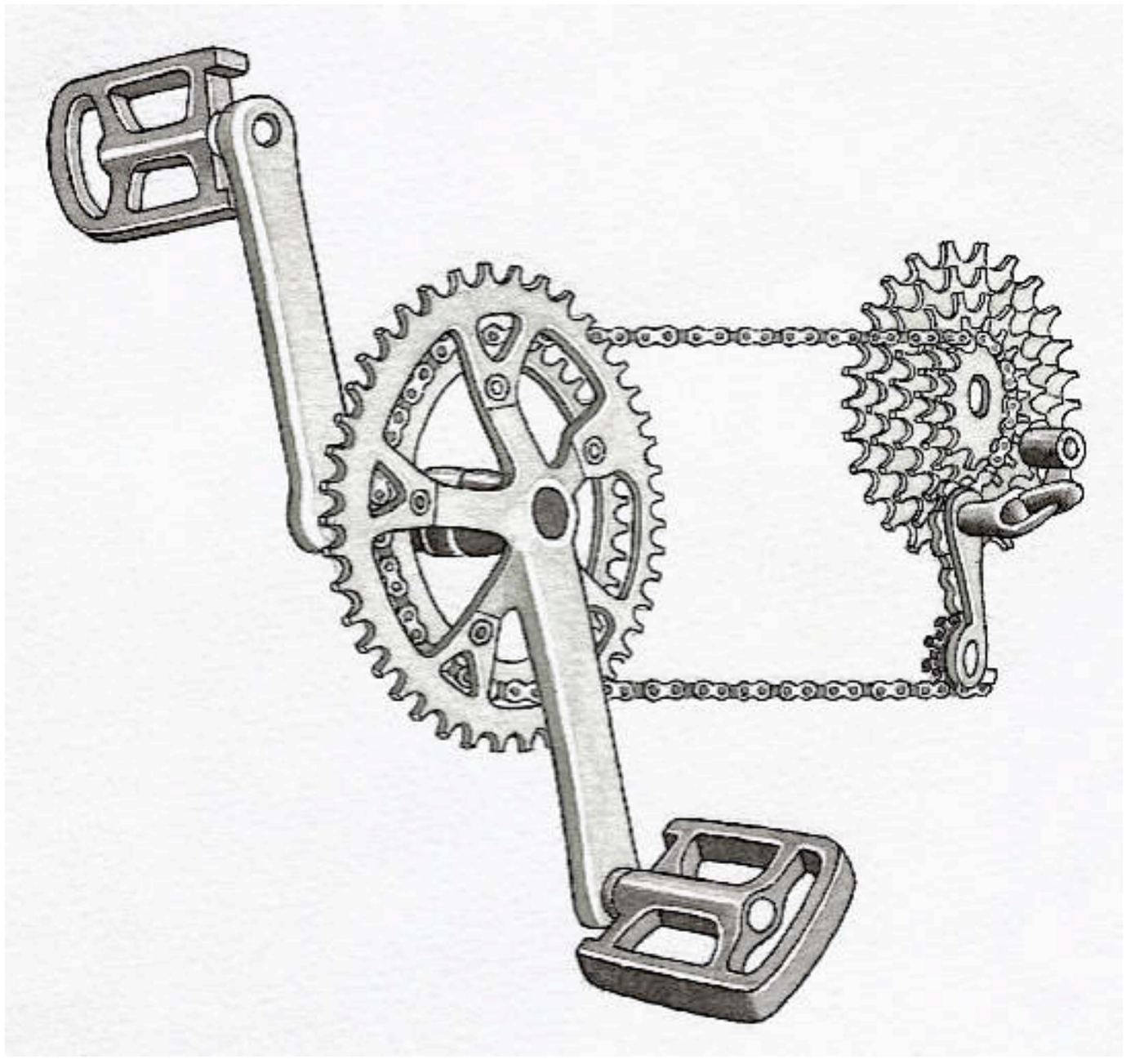
- Add variables to the domains





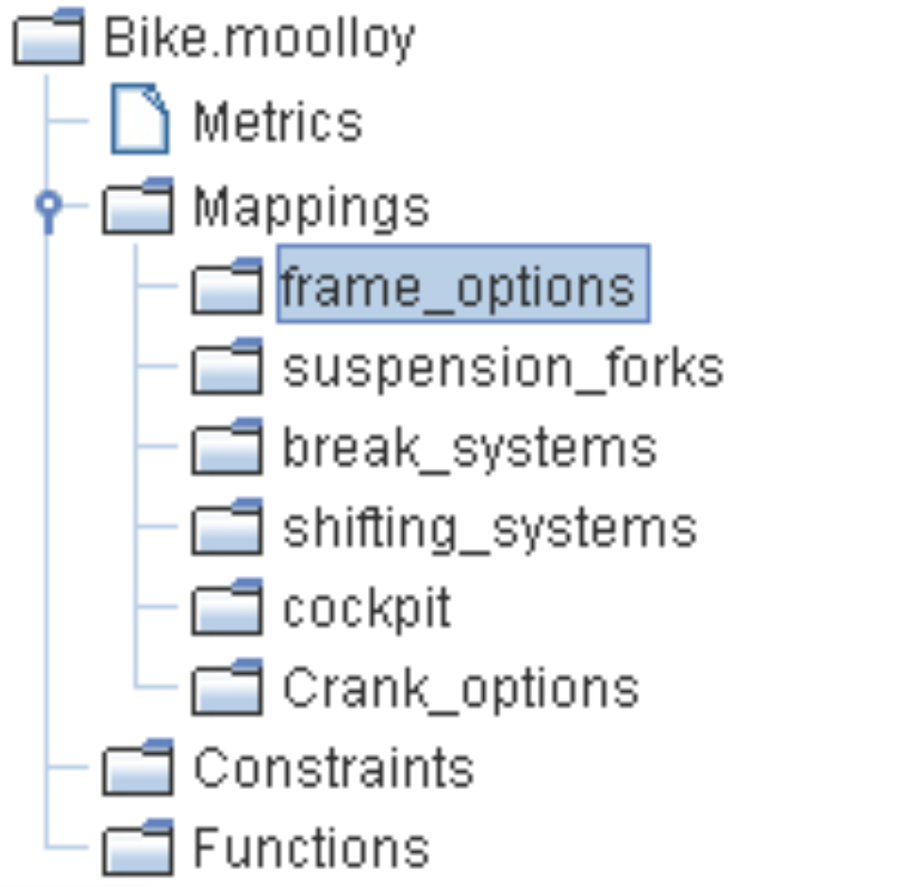
3 Decisions in 1 Domain

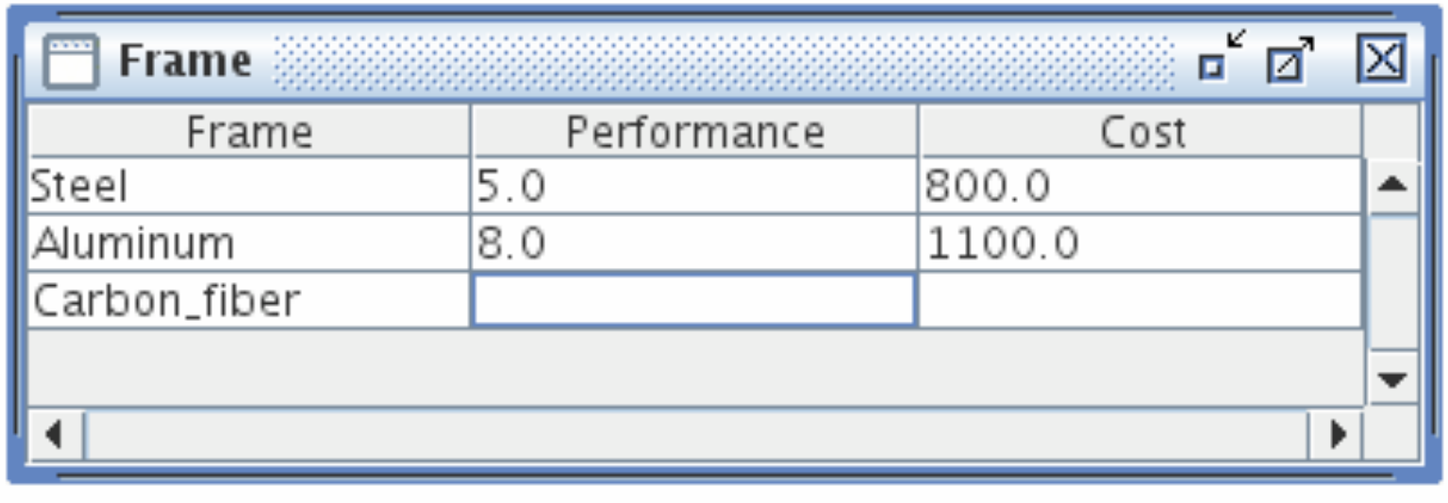




Metric Values

- Add variables to the domains
- Enter metric values for assignments

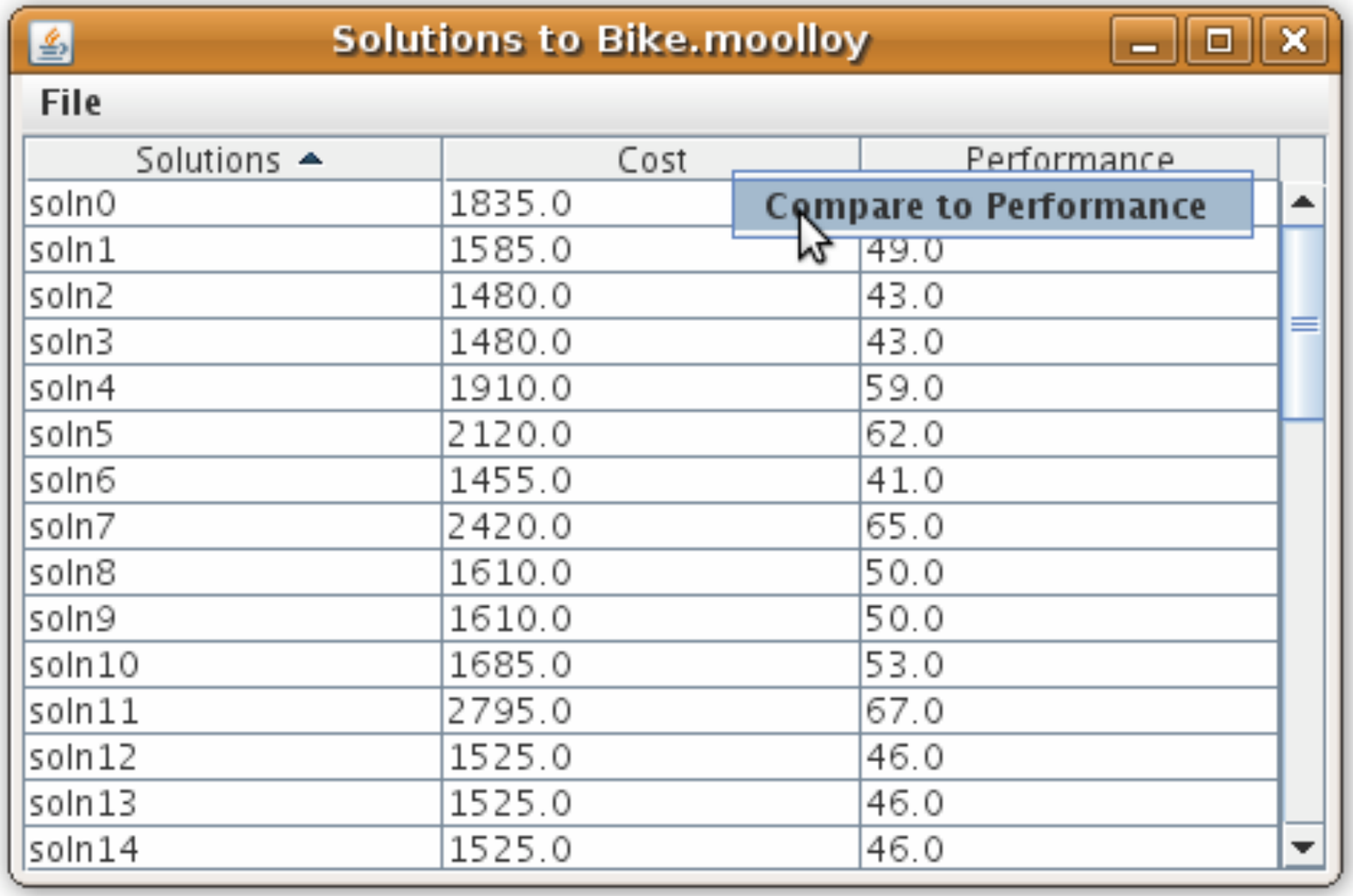




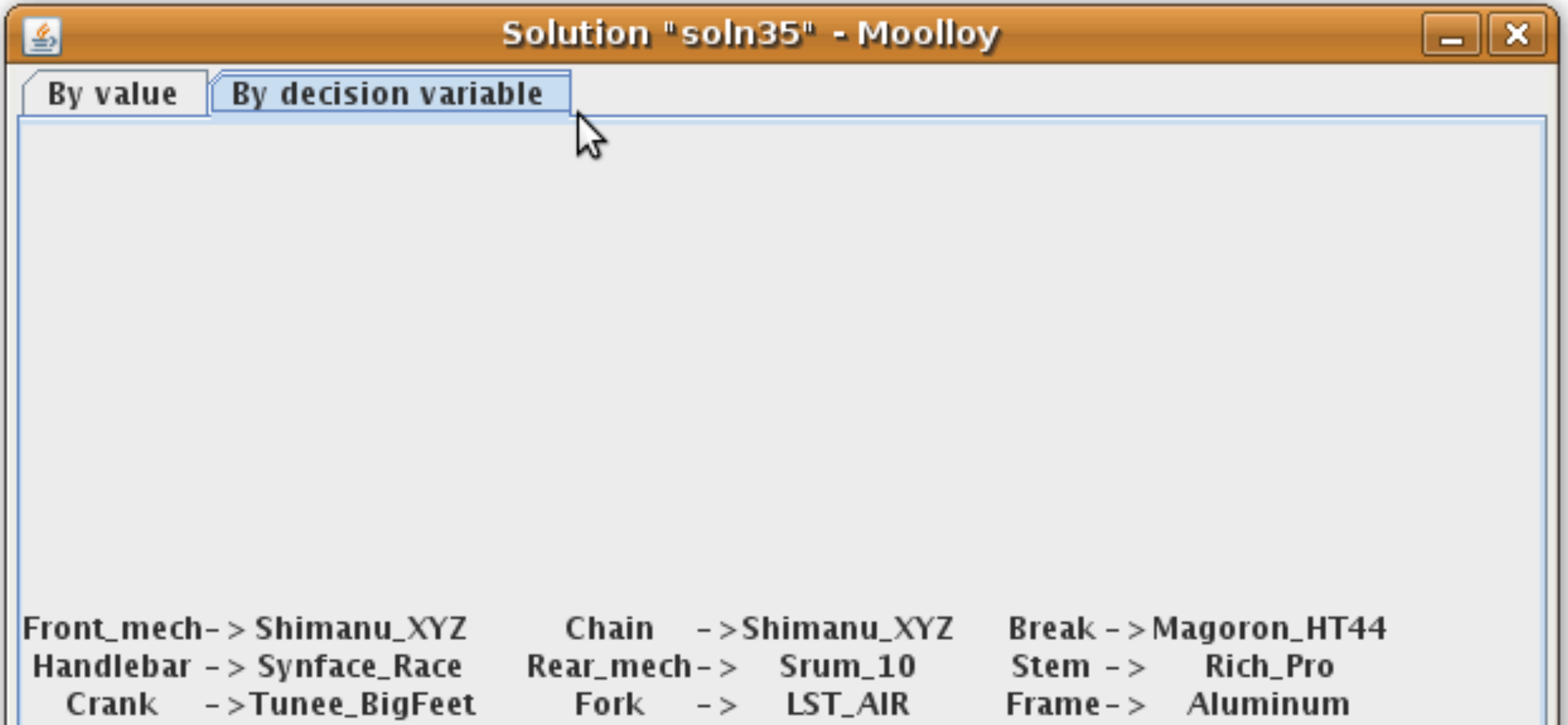
Solve!

List of Pareto Solutions

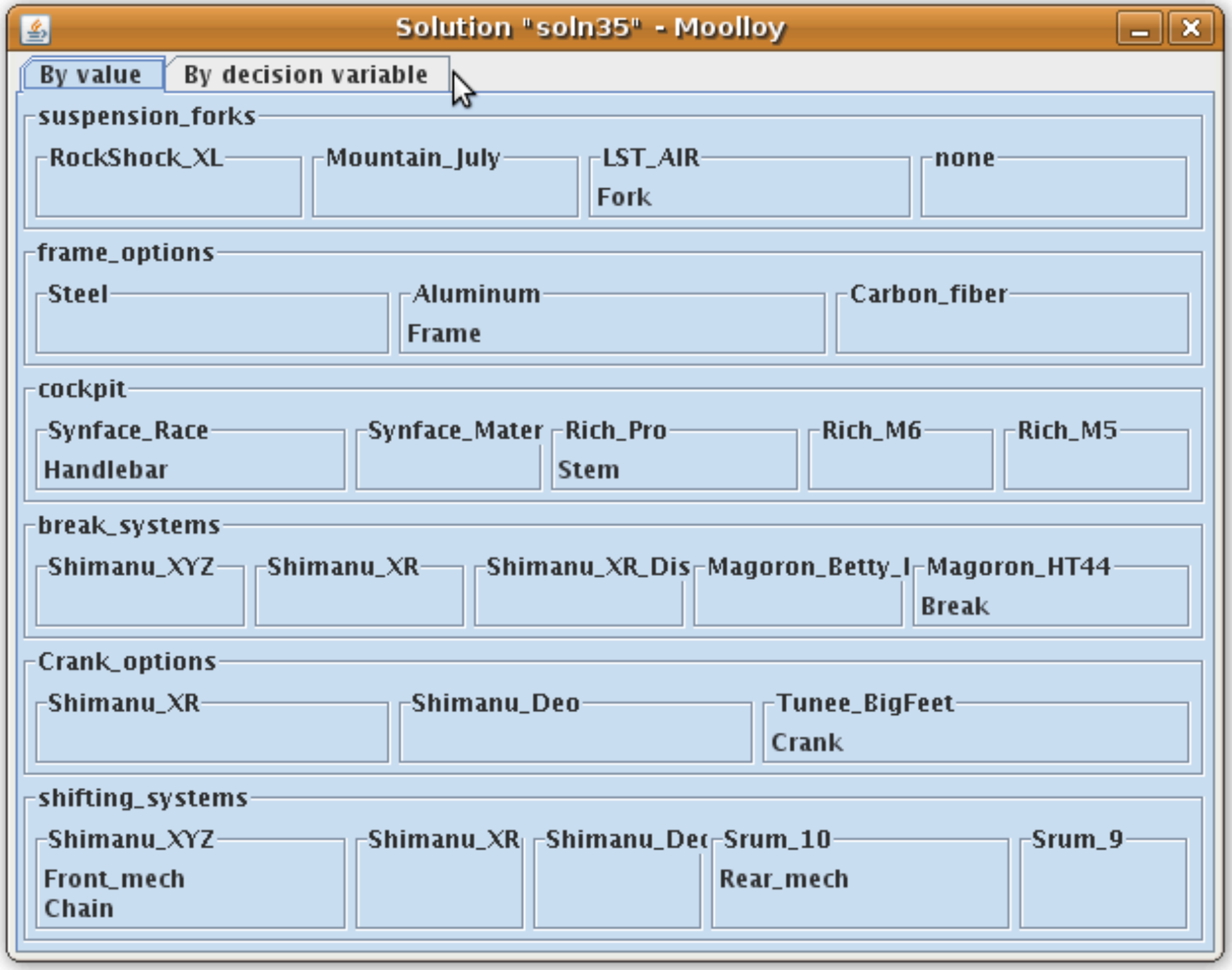
- Each solution in list can be inspected
- Can plot 2D Pareto front



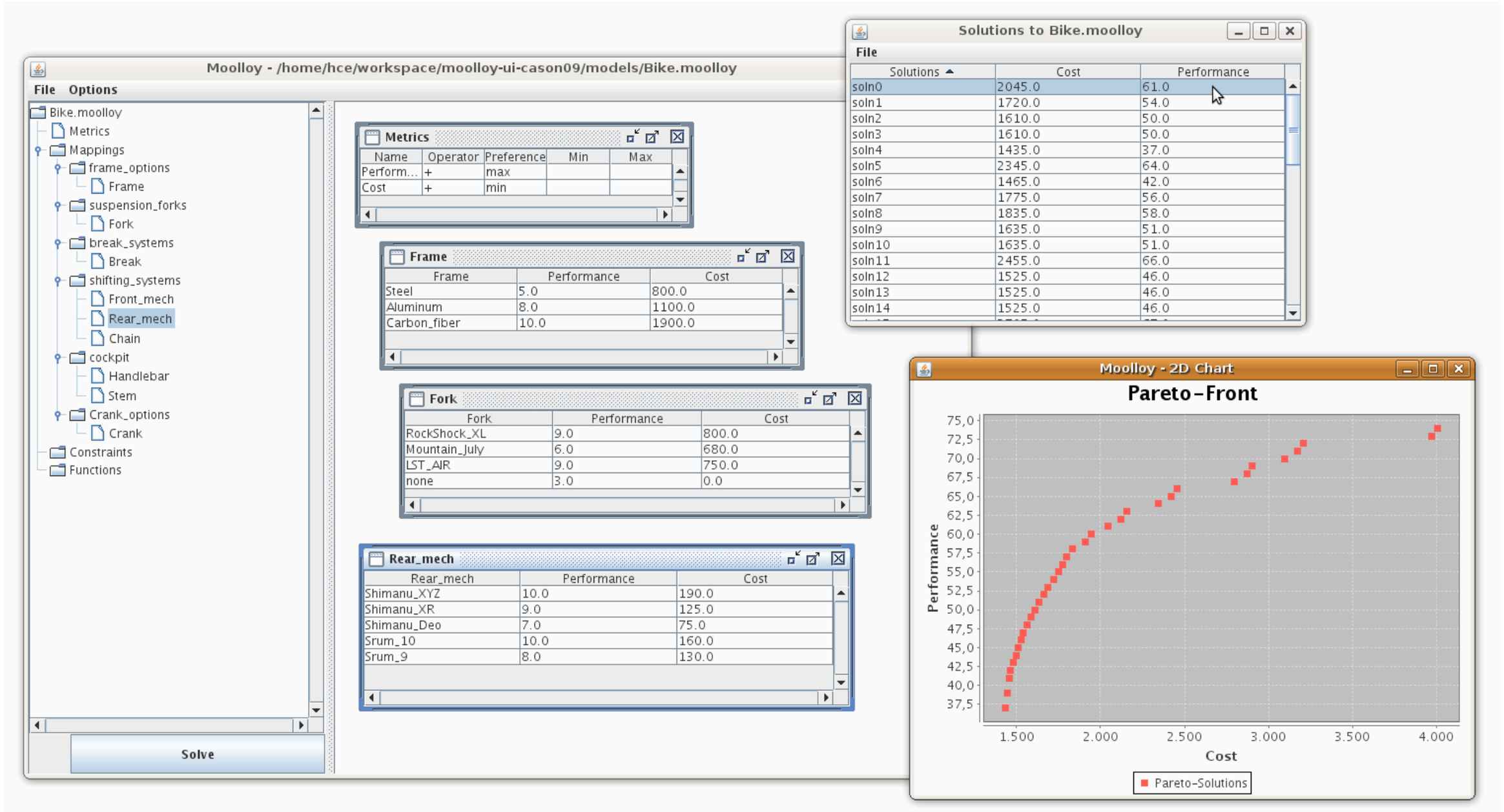
Inspecting a single solution



Inspecting a single solution



UI Overview


Moolloy - /home/hce/workspace/moolloy-ui-cason09/models/Bike.moolloy
File Options
Bike.moolloy
Metrics
Mappings
frame_options
Frame
suspension_forks
Fork
break_systems
Break
shifting_systems
Front_mech
Rear_mech
Chain
cockpit
Handlebar
Stem
Crank_options
Crank
Constraints
Functions
Solve
Metrics
Name Operator Preference Min Max
Perform... + max
Cost + min
Frame
Frame Performance Cost
Steel 5.0 800.0
Aluminum 8.0 1100.0
Carbon_fiber 10.0 1900.0
Fork
Fork Performance Cost
RockShock_XL 9.0 800.0
Mountain_July 6.0 680.0
LST_AIR 9.0 750.0
none 3.0 0.0
Rear_mech
Rear_mech Performance Cost
Shimanu_XYZ 10.0 190.0
Shimanu_XR 9.0 125.0
Shimanu_Deo 7.0 75.0
Srum_10 10.0 160.0
Srum_9 8.0 130.0
Solutions to Bike.moolloy
File
Solutions Cost Performance
soln0 2045.0 61.0
soln1 1720.0 54.0
soln2 1610.0 50.0
soln3 1610.0 50.0
soln4 1435.0 37.0
soln5 2345.0 64.0
soln6 1465.0 42.0
soln7 1775.0 56.0
soln8 1835.0 58.0
soln9 1635.0 51.0
soln10 1635.0 51.0
soln11 2455.0 66.0
soln12 1525.0 46.0
soln13 1525.0 46.0
soln14 1525.0 46.0
Moolloy - 2D Chart
Pareto-Front
Performance
75,0 72,5 70,0 67,5 65,0 62,5 60,0 57,5 55,0 52,5 50,0 47,5 45,0 42,5 40,0 37,5
1.500 2.000 2.500 3.000 3.500 4.000
Cost
Pareto-Solutions

Constraints

Constraints

- Three major types of constraints
  i. *Assignment constraints*
  ii. *Metric constraints*
  iii. *Functional constraints*

- *i)* , *ii)* are needed even for simple problems
- *iii)* more important for fine adjustments
  - (not yet supported by Moolloy UI)

Expressing Constraints

- Two ways of expressing constraints

  1) *Implicit*: using features built-in the UI

  2) *Explicit*: using a declarative expression language

Implicit Constraints

- Metric constraints

Metrics

Name	Operator	Preference	Min	Max
Performance	+	max	8	
Cost	+	min		1000

- Assignment constraints

Frame

Frame	Performance	Cost
Steel	5.0	800.0
Aluminum	8.0	1100.0
Carbon_fiber		

frame_options

frame_options	MinVars	MaxVars
Steel		
Aluminum		
Carbon_fiber		

Explicit Constraints

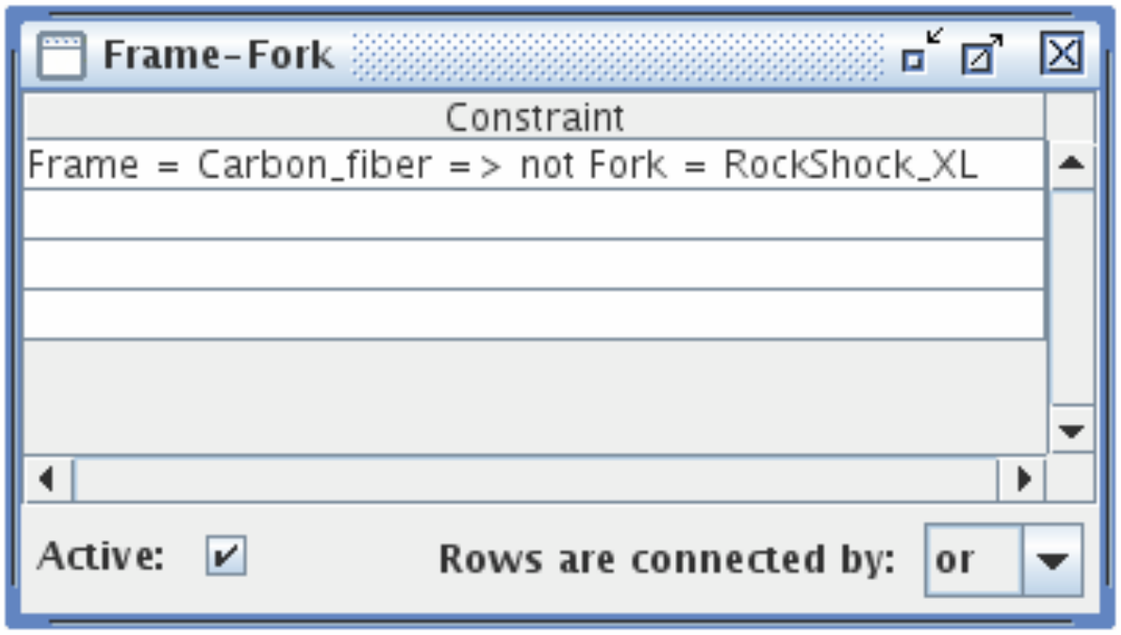
- Grammar for simple expression language:

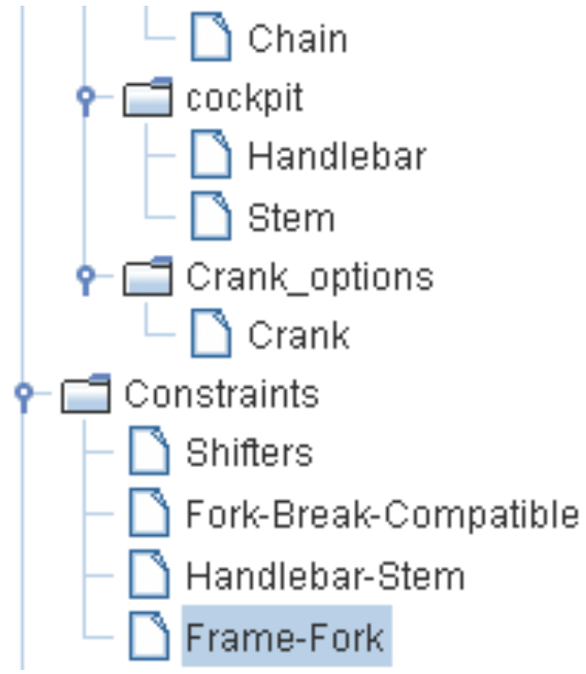
```
constraint ::= '(' constraint ')'
     | '@' classId
     | negationOp constraint
     | constraint logicOp constraint
     | formula
     | logicConst

  negationOp ::= 'not' | '!'

logicOp ::= 'and' | '&'
   | 'implies' | '=>'
   | 'iff' | '<=>'
   | 'or' | '|'

logicConst ::= 'true' | 'false'

formula ::= expr formulaOp expr
   | varId formulaOp (varId | valueId)
   | valueId formulaOp varId
```

```
formulaOp := '=' | '<' | '>' | '<=' | '>='

  expr ::= '(' expr ')'
     | expr exprOp expr
     | sumExpr
     | multExpr
     | intConst

exprOp ::= '*' | '/' | '%' | '+' | '-'

sumExpr ::=
   '$sum(' metricId ',' domainId ','
           valueId ',' number ')'

multExpr ::=
   '$mult(' metricId ',' domainId ','
            valueId ',' number ')'

intConst ::= '$'number
```

An Explicit Constraint

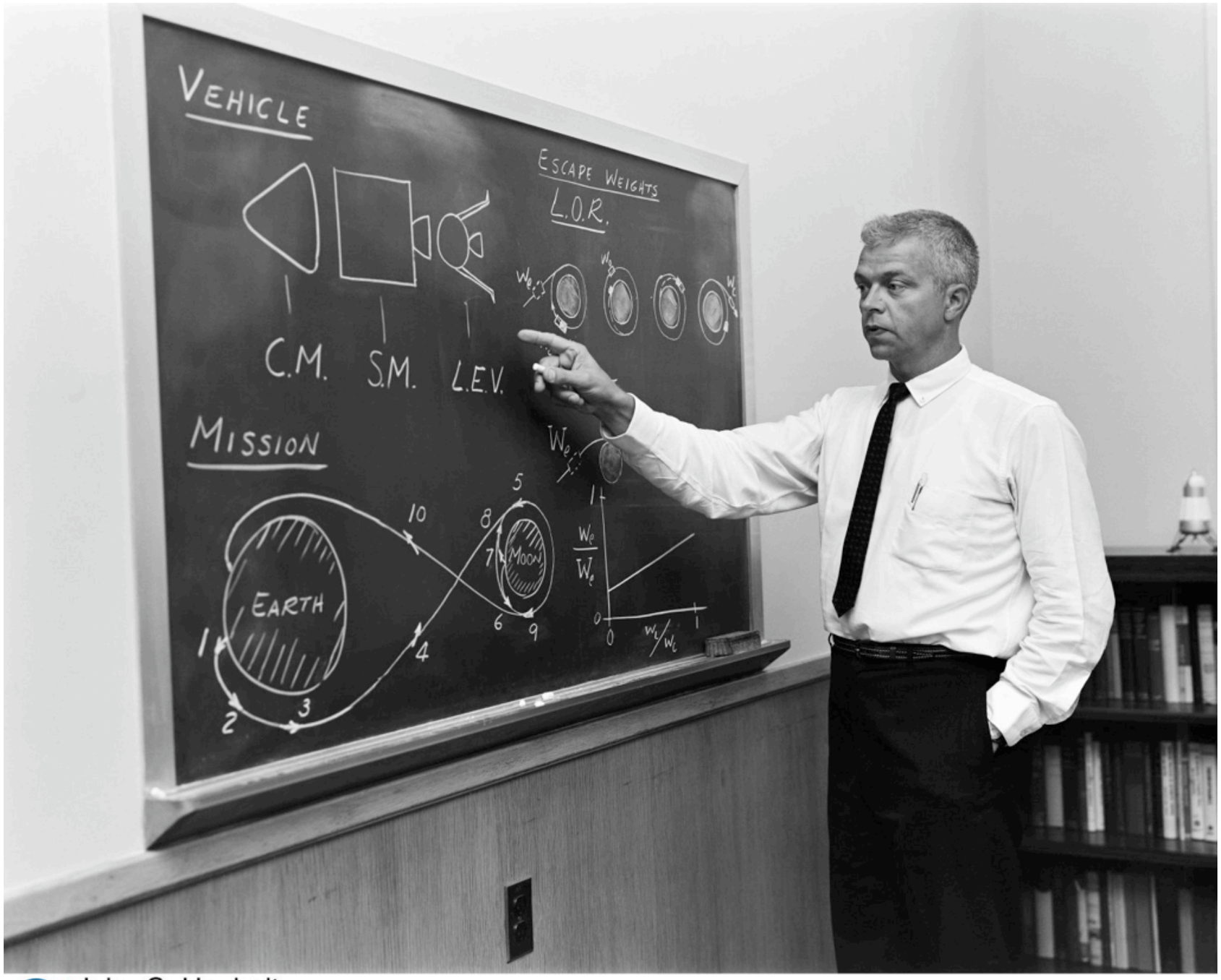
- **`Frame=Carbon_fiber => not (Fork=Shocks)`**





User Case Studies

Apollo Revisited

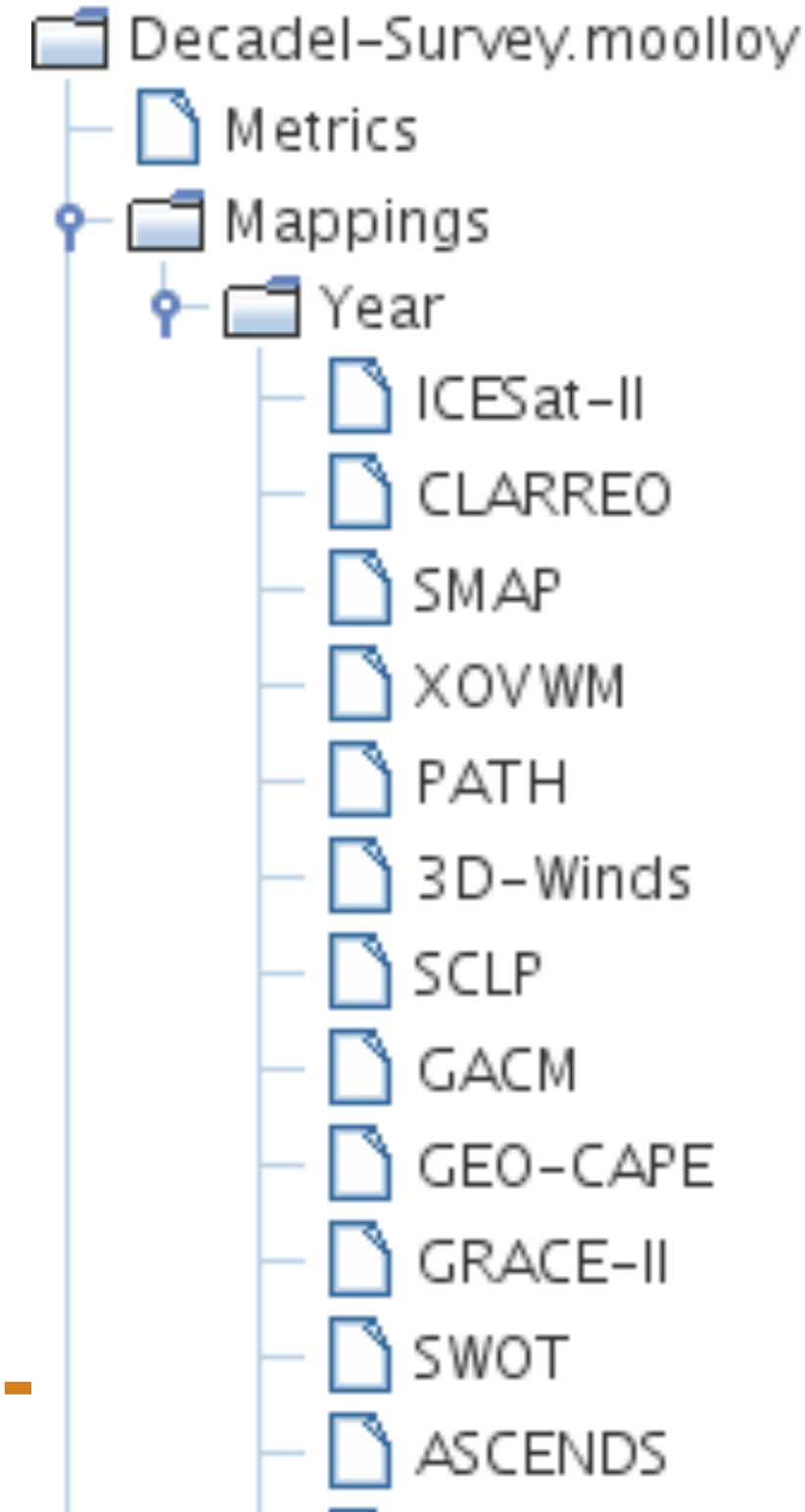


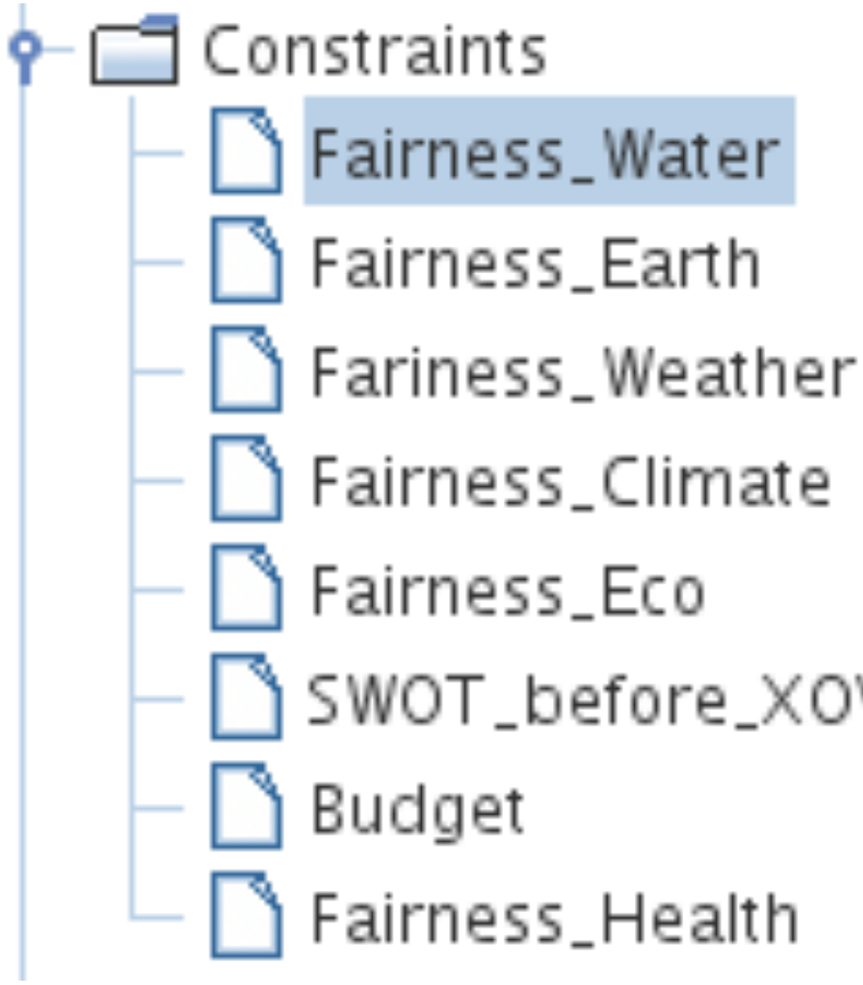
John C. Houbolt
NASA Langley Research Center 7/10/1962 Image # EL-2000-00424

Production Planning

Decadel Launch Schedule

Decisions & Constraints





Metrics

Metrics

Name	Operator	Preference	Min	Max
Health	+	max		
Water	+	max		
Eco	+	max		
Weather	+	max		
Earth	+	max		
Climate	+	max		
Cost	+	none		

Future Work

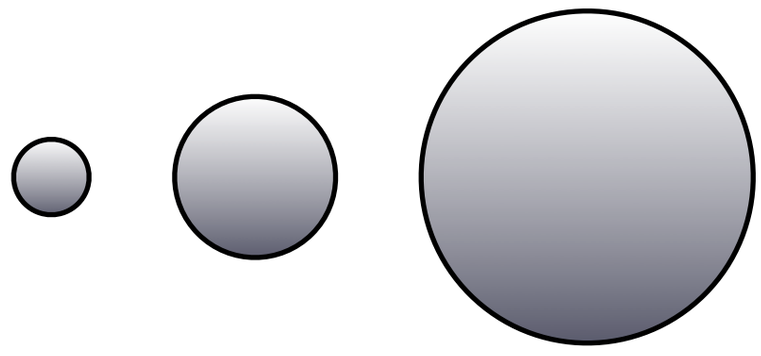
- Enhance UI with Schematic Tables
  - For expressing explicit constraints.
  - For inter-dependent metric functions.
  - [Edwards @ OOPSLA/Onward 2007]

- Visualizing Pareto-fronts in >2 dimensions
  - Discussion in workshop tomorrow afternoon

- Usability studies

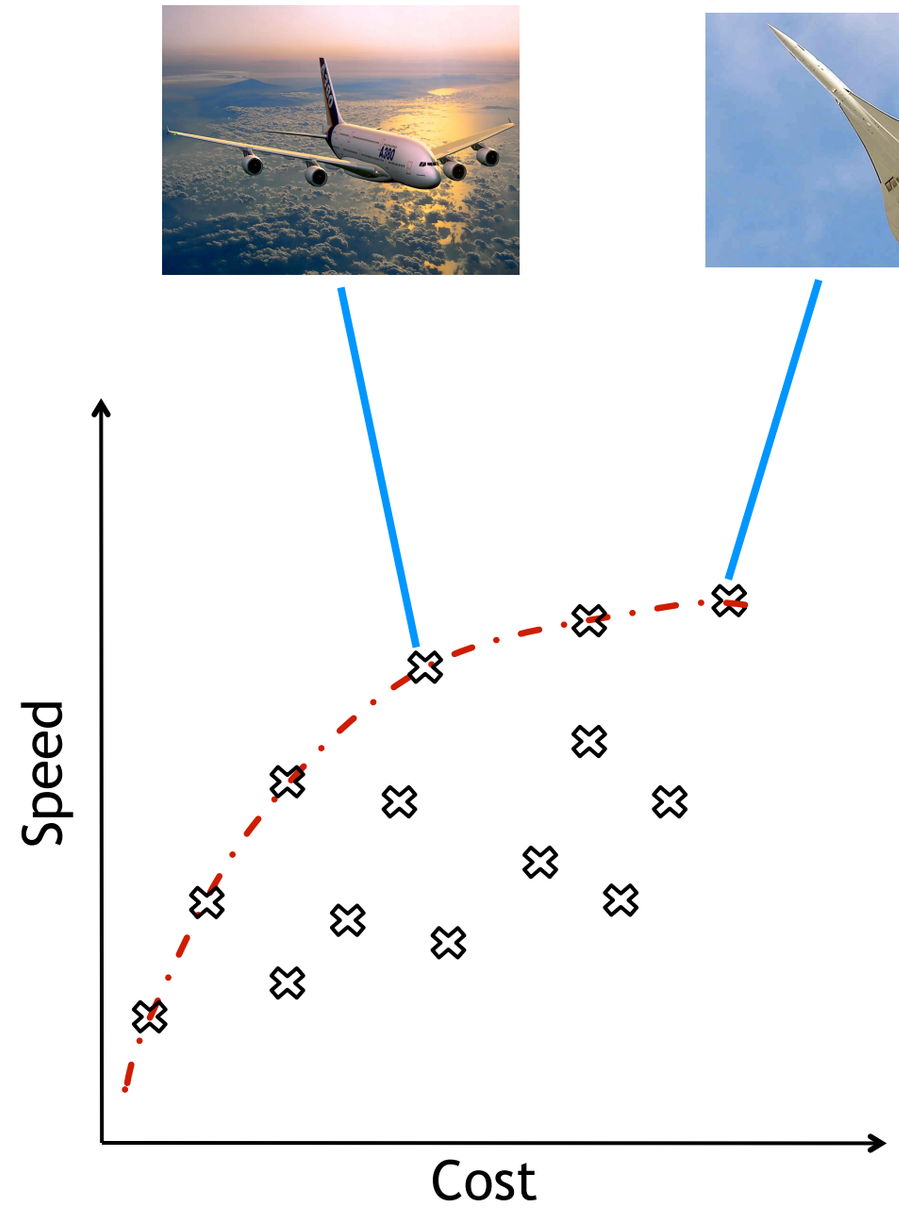
Conclusions

- Multi-objective optimization is not a hard idea.
  - Over 100 years old.
  - (But, designing algorithms & UIs is challenging.)
- It could be more widely used with better tools.
  - Potential users:



- http://sdg.csail.mit.edu/moolloy

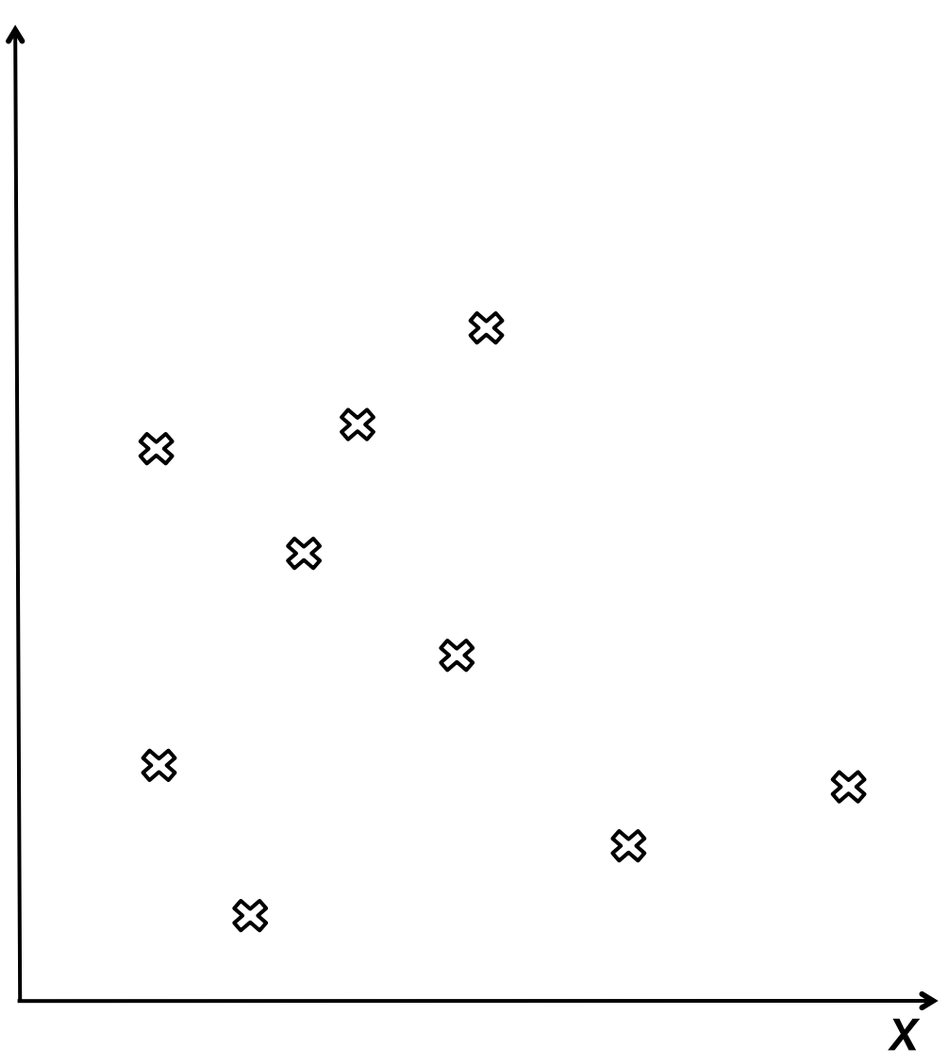
http://sdg.csail.mit.edu/moolloy



Extra Slides

Guided Improvement

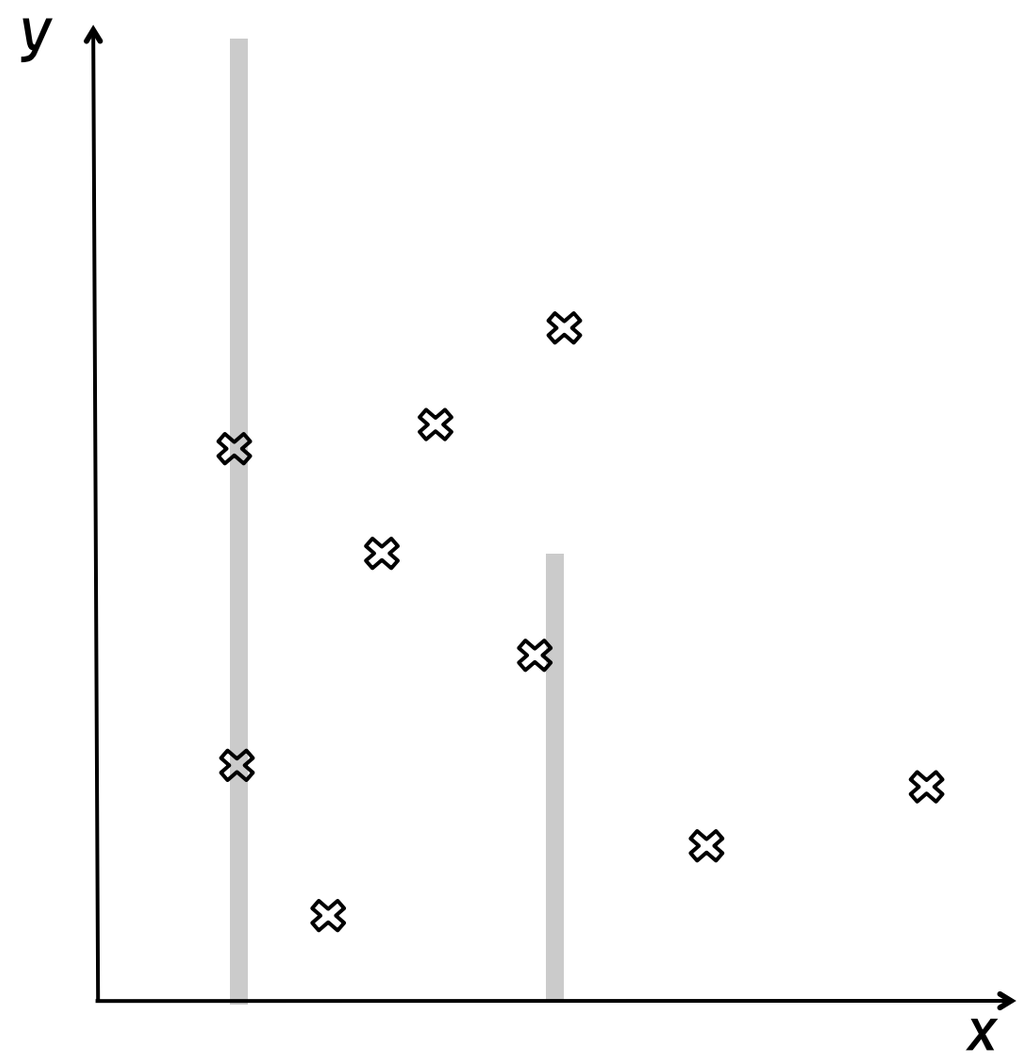
maximize *x*
maximize *y*



Output:

Guided Improvement

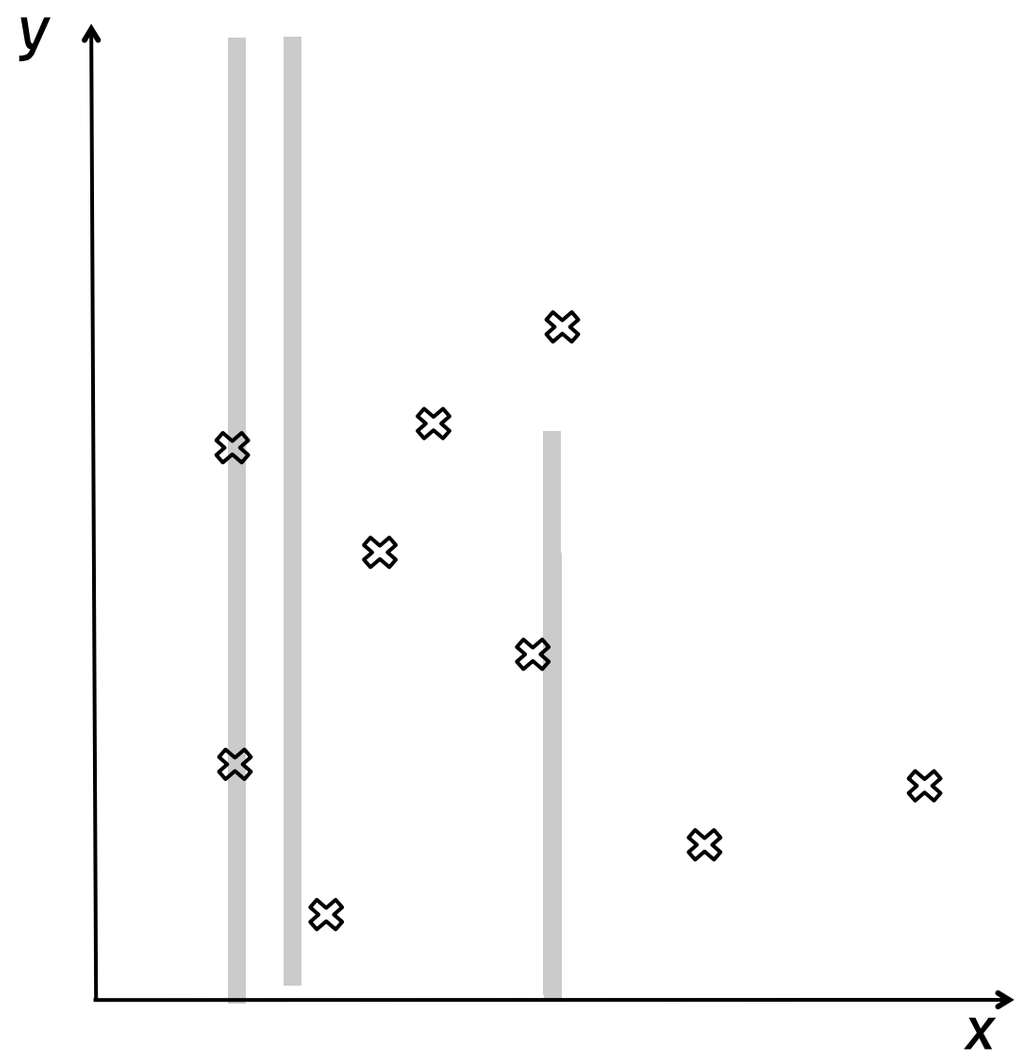
maximize *x*
maximize *y*



Output:

Guided Improvement

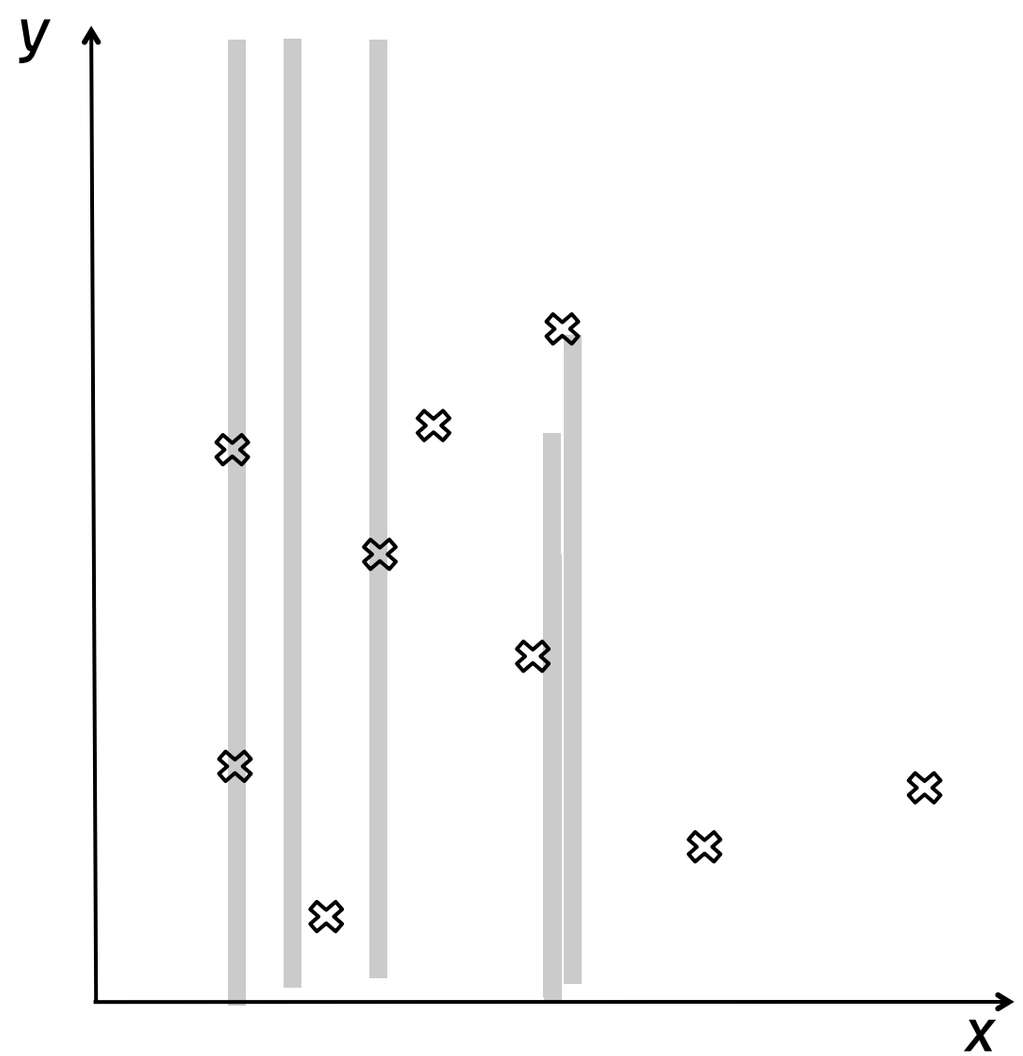
maximize *x*
maximize *y*



Output:

Guided Improvement

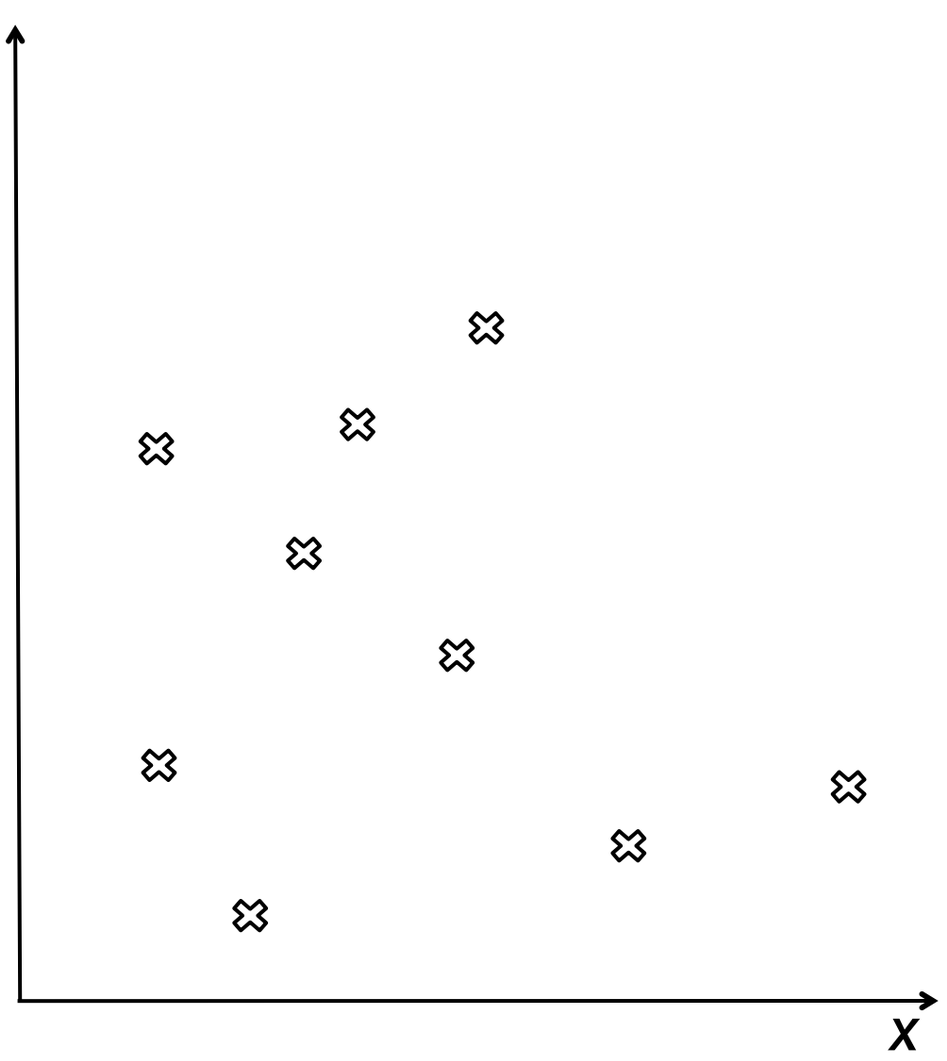
maximize *x*
maximize *y*



Output:

Guided Improvement

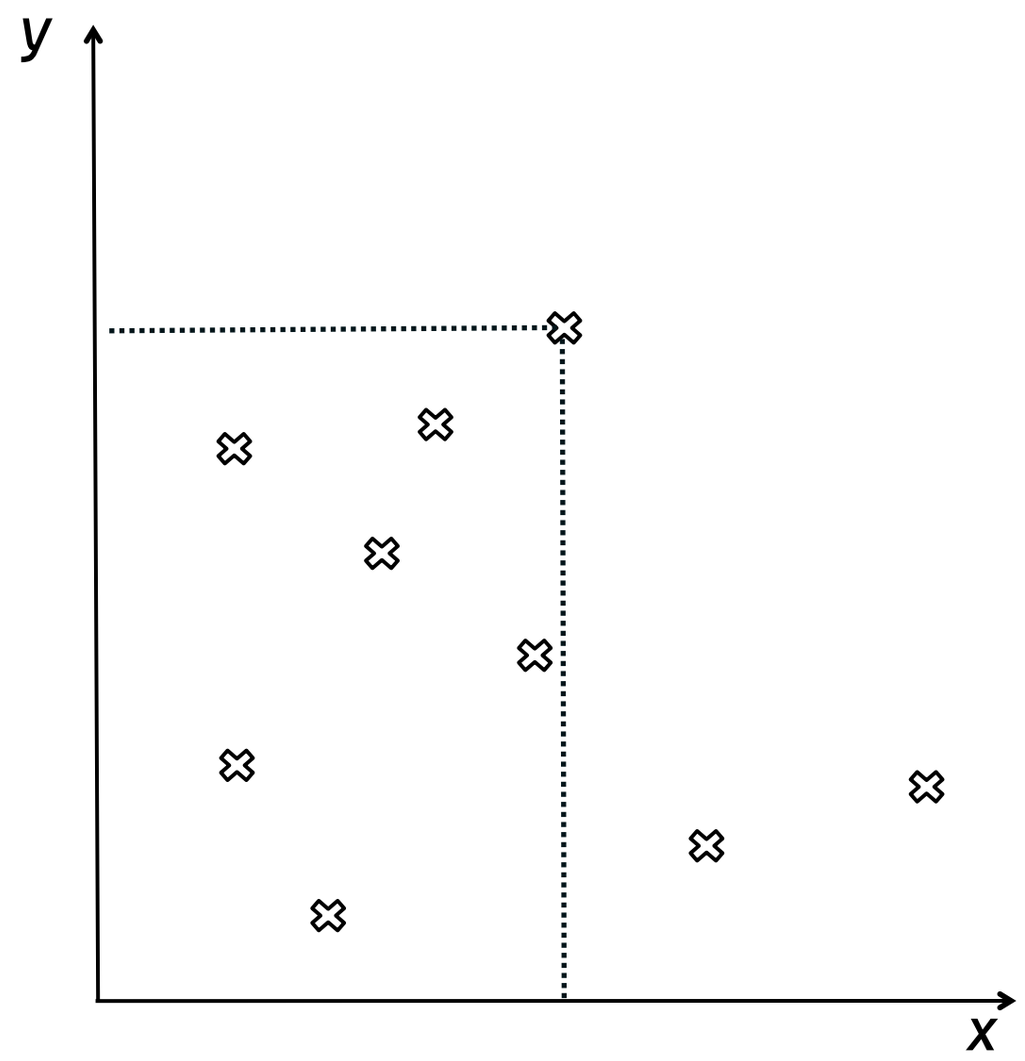
maximize *x*
maximize *y*



Output: S1, S2,

Guided Improvement

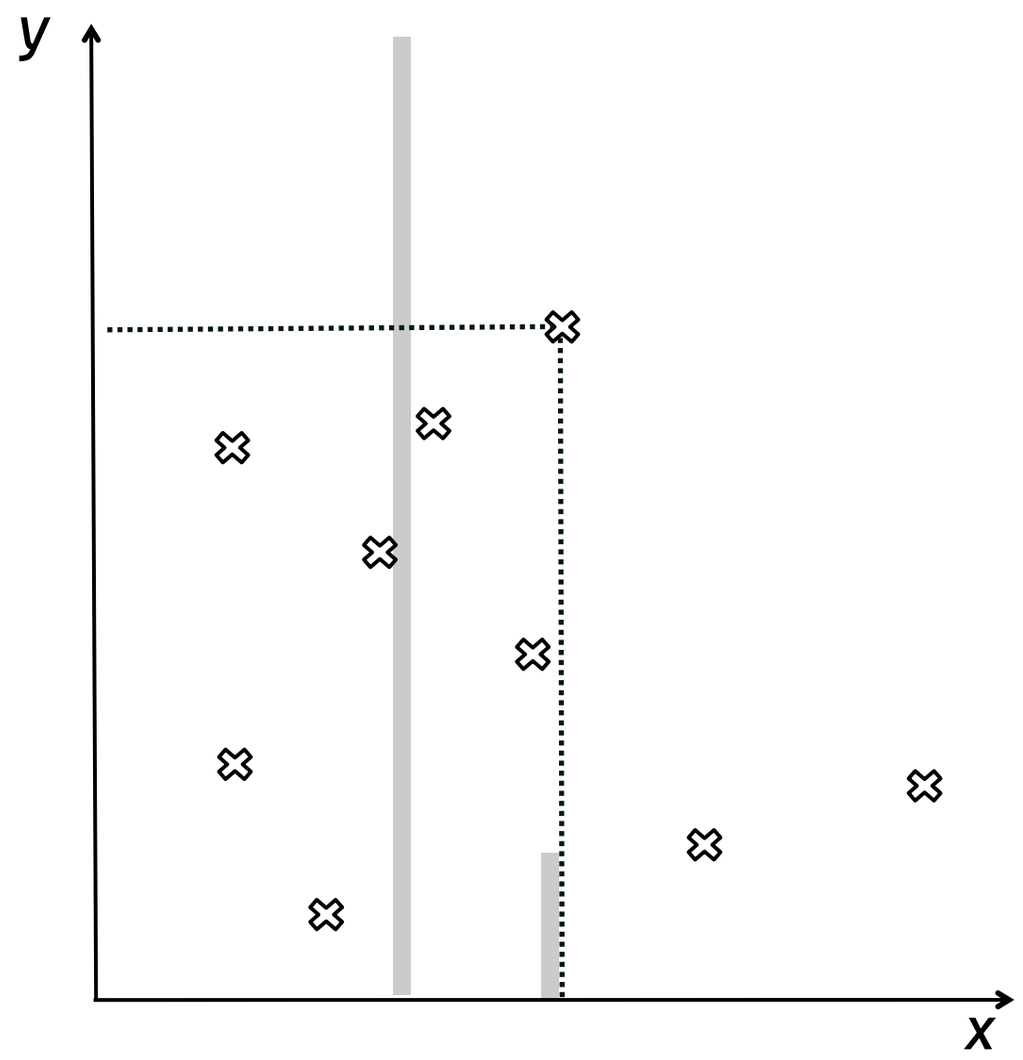
maximize *x*
maximize *y*



Output:S1, S2,

Guided Improvement

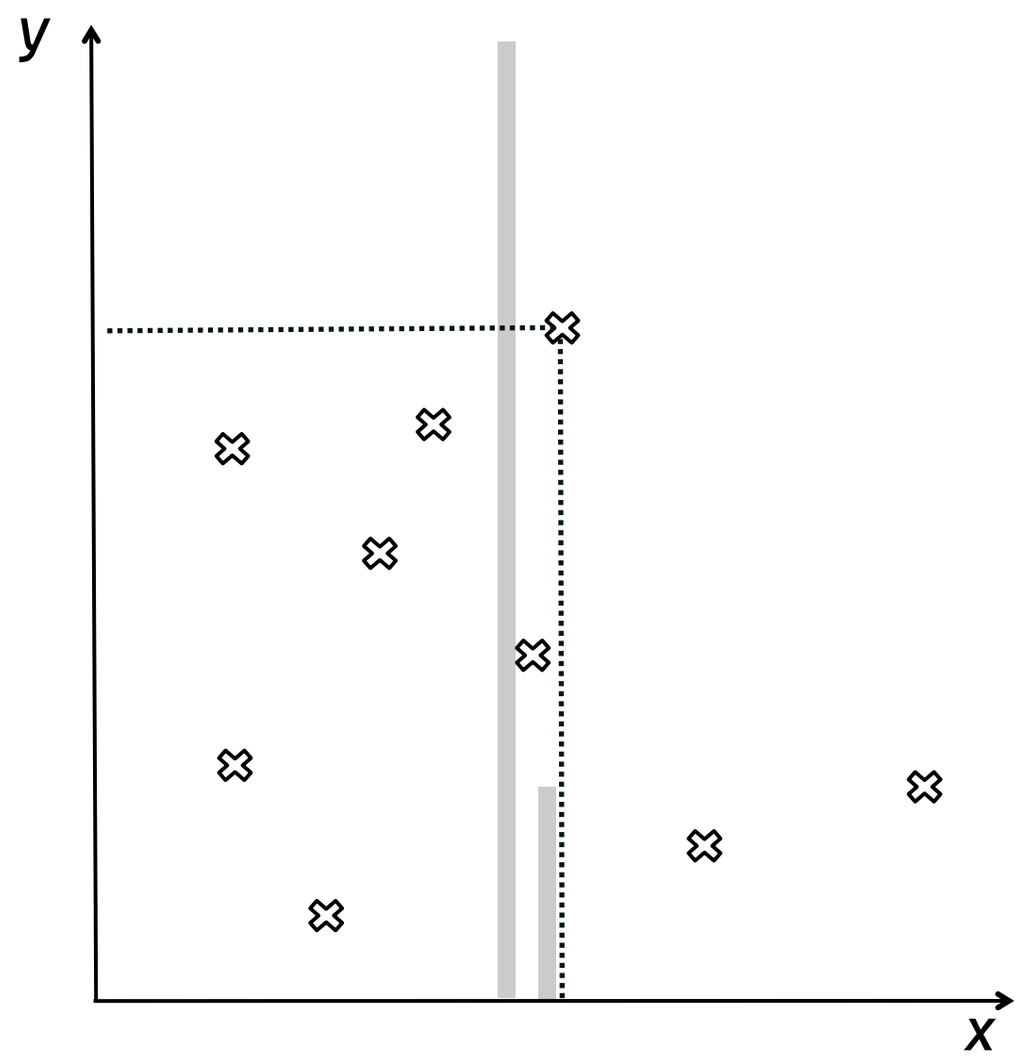
maximize *x*
maximize *y*



Output:S1, S2,

Guided Improvement

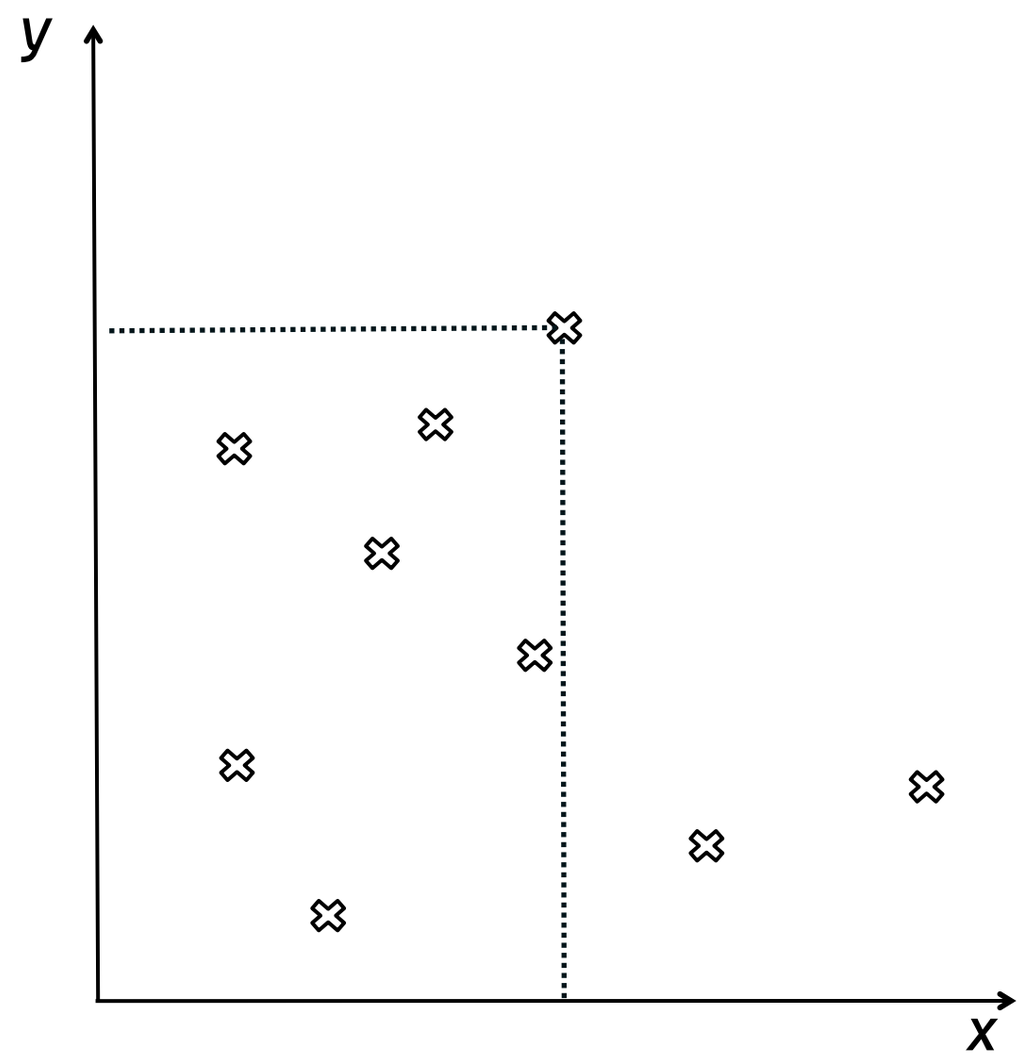
maximize *x*
maximize *y*



Output: S1, S2,

Guided Improvement

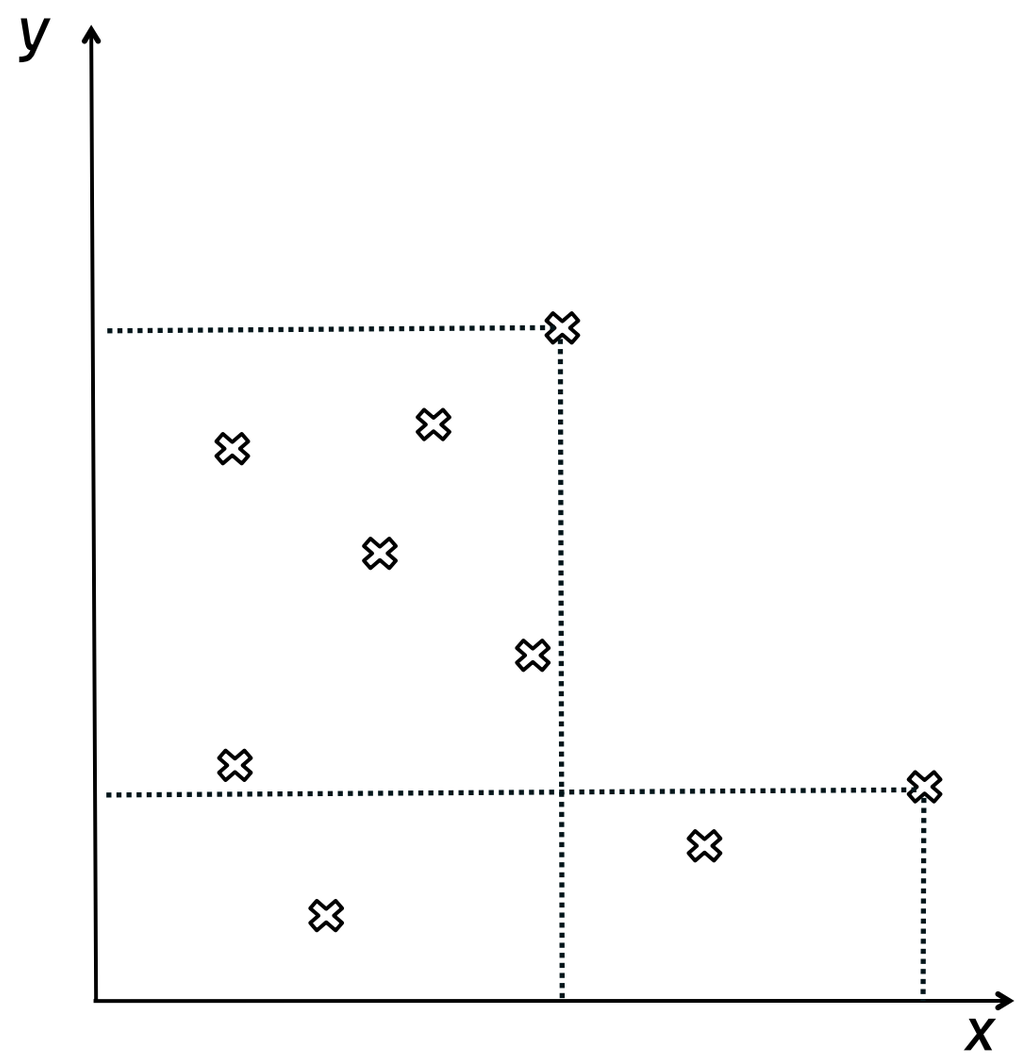
maximize *x*
maximize *y*



Output:S1, S2, S3,

Guided Improvement

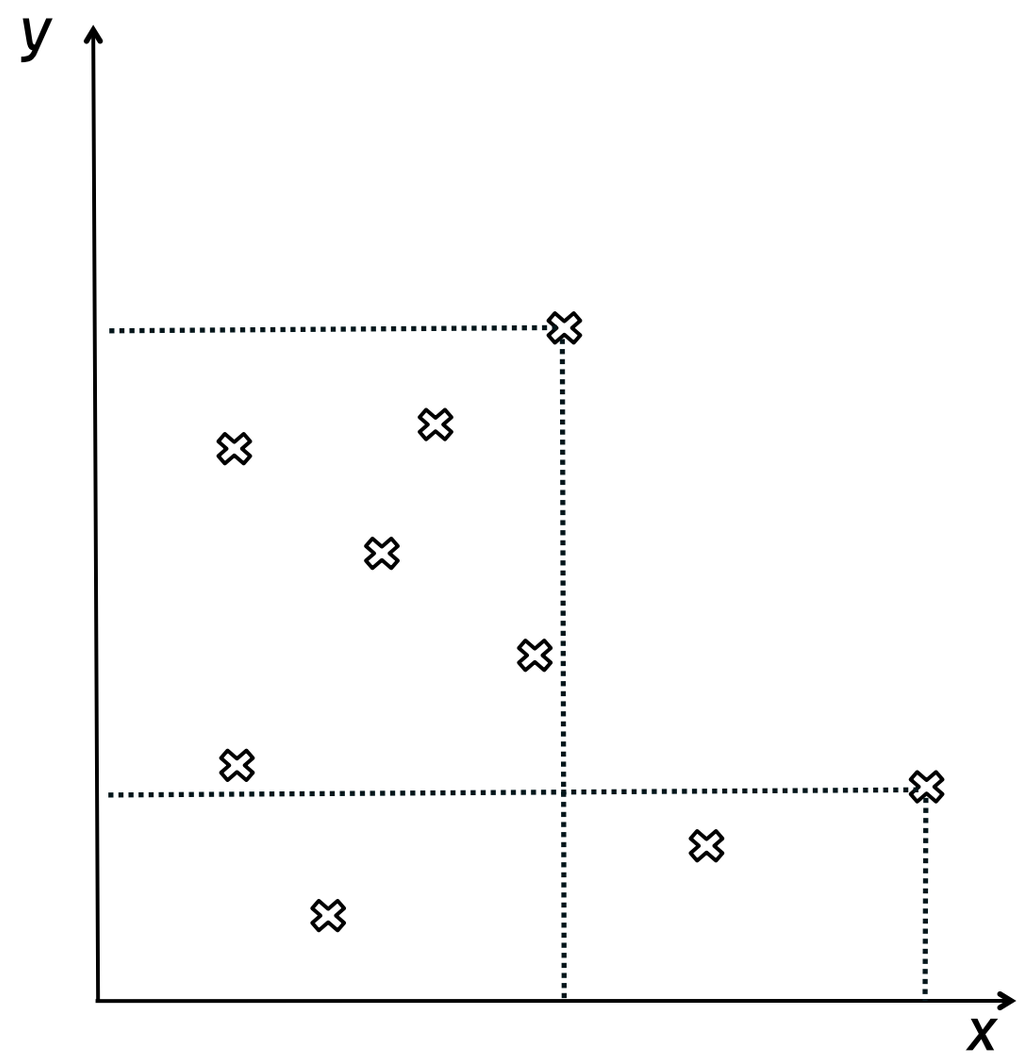
maximize *x*
maximize *y*



Output:S1, S2, S3,

Guided Improvement

maximize *x*
maximize *y*



Output: S1, S2, S3, Termination

Concorde SST

Motivation

- Finding *optimal solutions* is a key-problem
- Relevant for many areas (Science & Industry)
- Decades of research
  - Multitude of solving methods
  - Multitude of tools

Motivation

- Most tools solve Single-Objective problems
  - We optimize <u>one</u> criteria (e.g. *Cost*)

- Insufficient for many real-world applications
  - E.g. minimize *Cost* and maximize *Performance*

Multi-Objective Optimization (MOOP)

- Finding optimal solutions for multiple objectives

Motivation

- Solving MOOPs is hard
  - Sophisticated algorithms
  - Heuristic approaches
  - *(not covered here)*

- In this talk: How do we formulate MOOPs?
  - Make MOOP accessible to non-experts
  - Make MOOP as easy as working with Spreadsheets

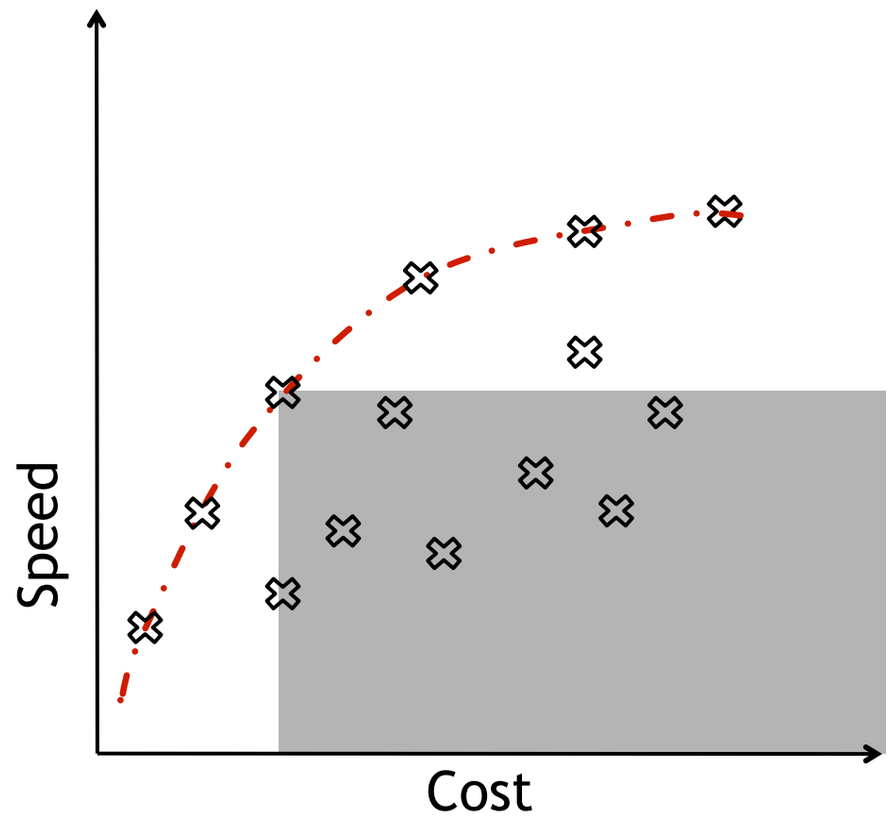
Example MOOP

- Compose bicycle from custom-made parts
  - Frame: steel, aluminum, …
  - Forks: RockShock XL, LST Air, …
  - Front mech: Srum 9, Srum 10, Shimanu XR, …
  - Rear mech: Srum 9, Srum 10, Shimanu XR, …
  - Chain: Srum 9, Srum 10, Shimanu XR, …

- Objectives / Metrics:
  - minimize *Cost*
  - maximize *Performance*

Case Study

- Expect solutions with
  - High *Cost* & high *Performance*
  - Low *Cost* & low *Performance*
  - Medium *Cost* & medium *Performance*
- Some solutions *dominate* other ones
- Non-dominated solutions are *Pareto optimal*



Insight / Conjecture