Learning IV&V strategies (from approximate models)

marcus.s.fisher@nasa.gov
IV&V, NASA, USA
tim@menzies.us
WVU, USA

Promise2Prototype : Promise2006

HomePage | Site map | Index | What's new | Comments | Legal

Change settings/Logout | You are TimMenzies

2nd International Workshop on Predictor Models in Software Engineering
(PROMISE 2006)

Sunday September 24, 2006, Philadelphia, Pennsylvania USA

An ICSM 2006 workshop,http://icsm2006.cs.drexel.edu/

NEWS: : Papers accepted to PROMISE 2006 will be eligable for submission to a special issue of the Journal of Empirical Software Engineering on repeatable experiments in software engineering.

IMPORTANT: PROMISE 2006 gives the highest priority to case studies, experience reports, and presented results that are based on publicly available datasets, preferably lodged at the PROMISE repository.

http://promise.unbox.org/Promise2006

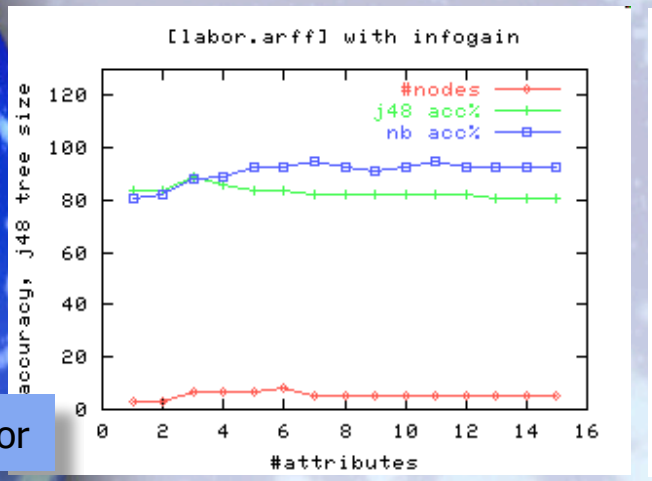
"design as search"

- Herbert Simon:
  - "Design = quintessential human activity"
- Allen Newell:
  - Cognition is a search for operators which we believe will take us towards our desired goals

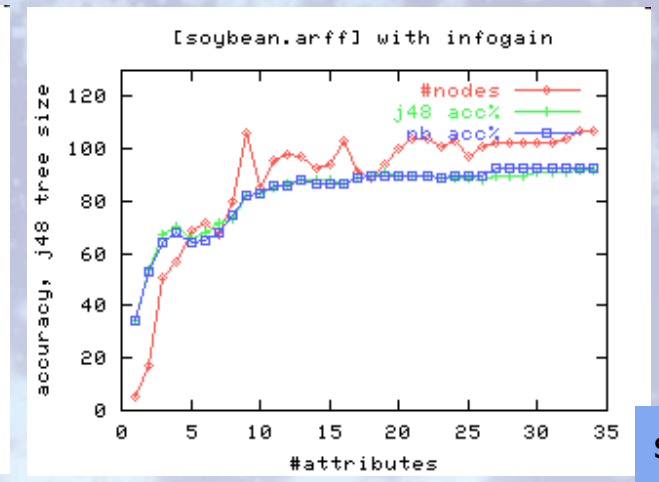
- Q: what if our beliefs are approximate?
  - I don't believe that you can always get rid of subjective judgments in these kinds of studies. -- Rick Kazman, Jan 6, 2006,10:53:47
- A: "Design" means doing lots of what-ifs.
  - Find consistent set(s) of beliefs a.k.a. "worlds"
  - What selects for worlds with results we want?

Surprisingly, don't need to explore all settings to all variables

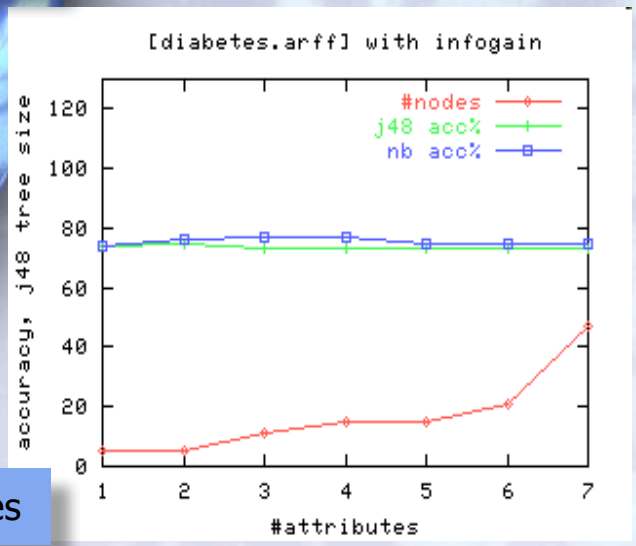
If sort attributes on "infogain" and learn using first N attributes then good theories with low N



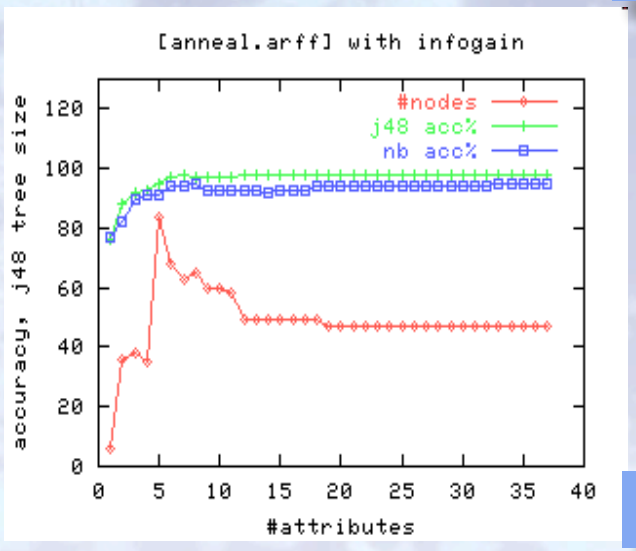
labor



soybean

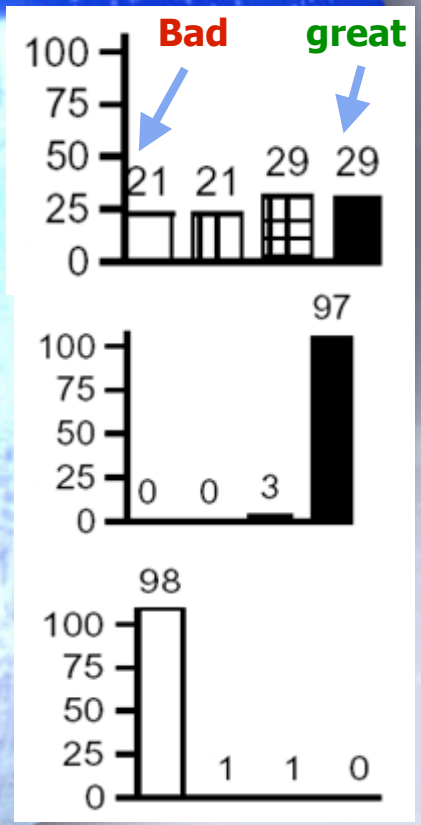


diabetes



anneal

So, we can "cheat"



Housing, baseline (% of housing types)

6.7 <= rm < 9.8 and 12.6 <= ptratio < 15.9

0.6 <= NOX < 1.9 and 17.16 <= lstat < 39

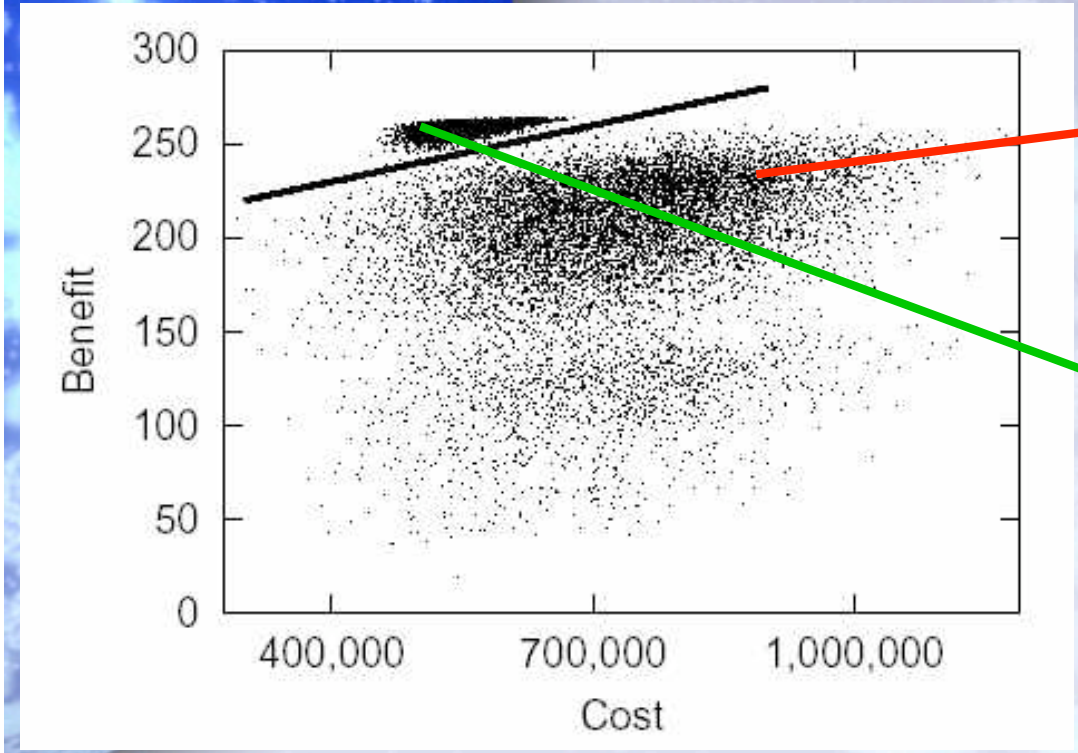
Method:

1. Stochastic sampling of lightweight notations
   - ❄ Explore all the what-ifs
2. Data mining to find the master variables

- Treatment" = policy
  - what to do
  - what to watch for
- TAR3
  - Seek attribute ranges that are often seen in "good"
  - and rarely seen in "bad".
- Treatment= constraints that changes baseline frequencies

A few variables are (often) enough

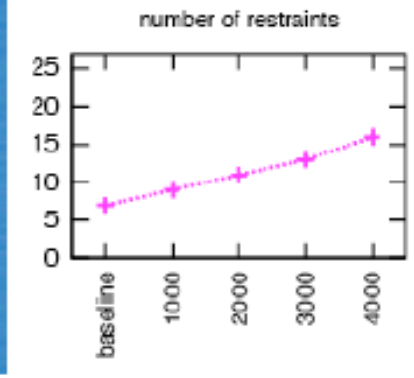
Case study 1



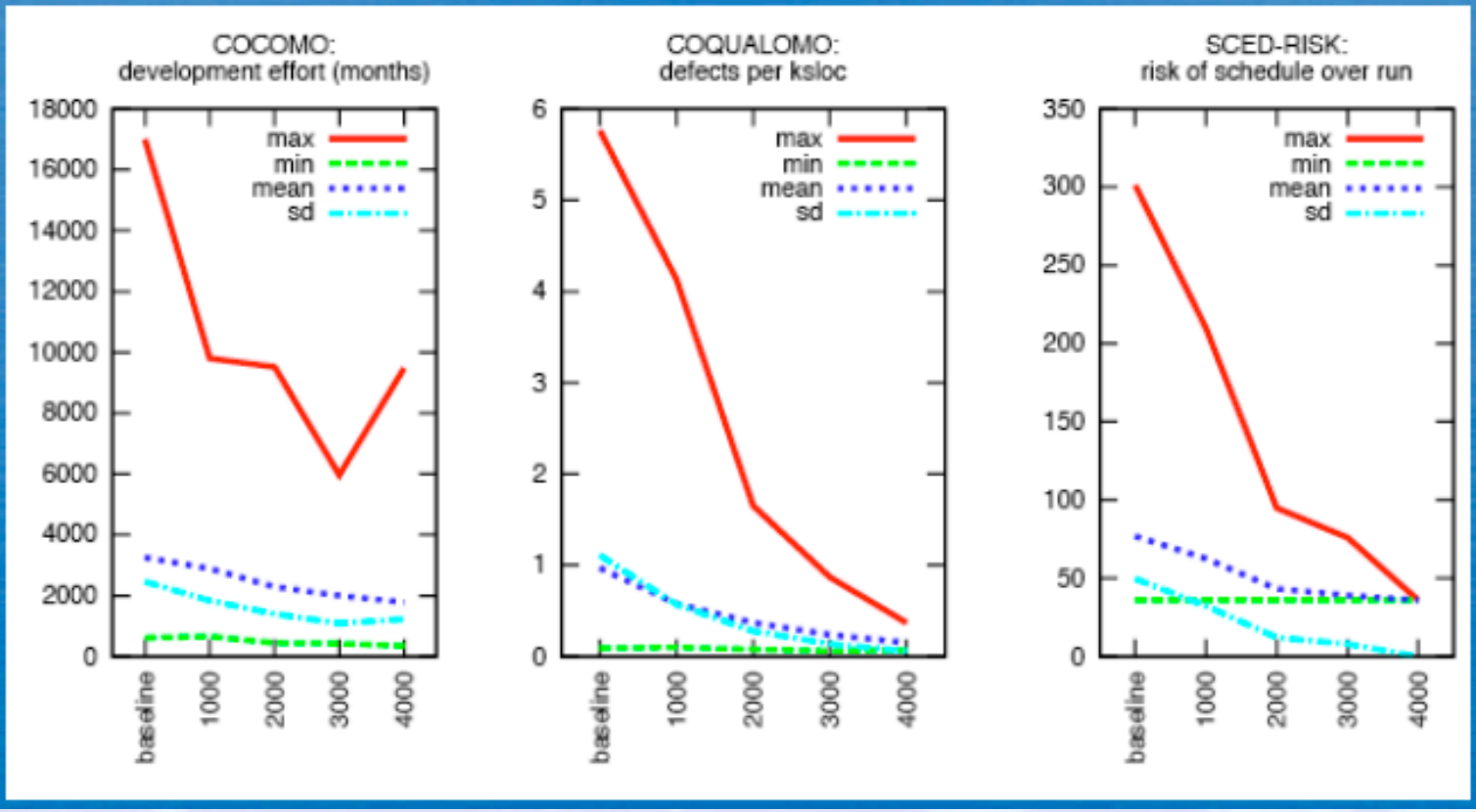
- JPL satellite design (Feather, Menzies 2002)
- 99 binary options.
  - Huge space of costs/benefits for those options
- TAR3 found 30 choices that collapsed options space
  - 66 choices that didn't matter

Case study 2: XOMO: Optimization of COCOMO-family models

- COCOMO:
  - effort estimation
- COQUALMO:
  - bugs introduced - bugs removed
- Madachy model:
  - how many dumb things are you doing today?
- Incremental optimization over 26 variables
- Case study: building autonomous systems



baseline	learned restraints			
	1000	2000	3000	4000
75 $\leq$ ksloc $\leq$ 125 rely = 5 prec = 1 acap = 5 aexp = 1 cplx = 6 ltex = 1 ruse = 6	sced=4 peer_reviews=5	pmat=5 pcap=4	tool=4 execution_testing- _and_tools=5	team=5 resl=5 automated- analysis=5



Case study 3 (SILAP)

Q: What most increases project errorPotential?

SILAP

- from DELPHI sessions with experienced NASA IV&V managers
- a network of weighted project factors
- E.g.

*function the(X) { return one (X) * all(X) }*

- *One:* project data
- *All:* DEPHI knowledge

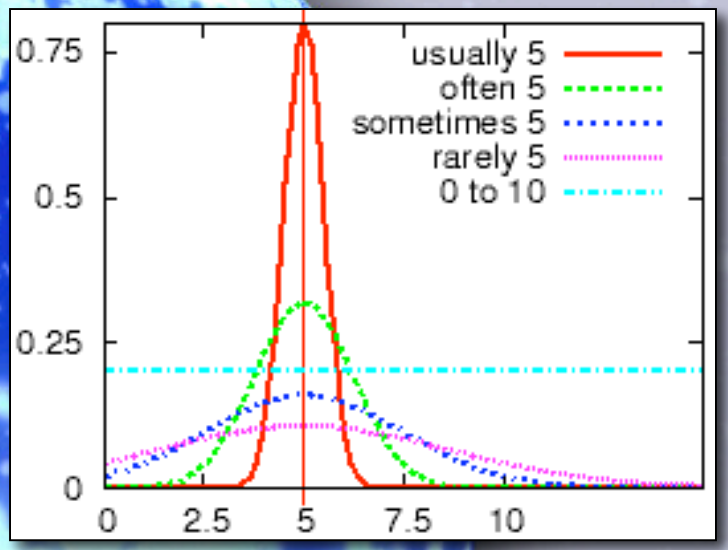
- E.g.

```
function development() {
    return the("experience") +
           the("organization") }
function software() {
    return  the("complexity") +
            the ("innovation") +
            the("softwareSize")
```

just a notation we made up one night

- Passes the "elbow test"
  - Domain experts elbow us out of the way …
  - … in their haste to fix some error.

SILAP models contain "hedges"



- *One* and *All* are defined using *hedges*; e.g.
  - One["complexity"] = usually 2
  - One["configManagement"] = sometimes 5;
  - One["defectTracking"] = rarely 3;
  - All["Experience"]= 0.8 to 0.9
  - All["Reuse"] = often 0.226
- Hedges define the spread (a.k.a. standard deviation) of a value:
  - usually(X) : mean=X, sd = 0.1*X
  - often(X) : mean=X, sd = 0.25*X
  - Sometimes(X) : mean=X, var = 0.5*X
  - Rarely(X) : mean=X, var = 0.75*X

just a notation we made up one night

Are there any stable conclusions in such space of maybes?

Commissioning SILAP

- Sampling studies:
  - Is Monte Carlo exploring enough of the model?
    - Distributions stabilize after 5000 samples
- Stability studies:
  - TAR3 is a stochastic search engine.
  - Do subsets of the data offer the same conclusions?
- Specialization studies:
  - Is there too much stability?
  - Do different software types yield different results?

Stability Studies (1)

- Run 5000 simulations
- Ten times,
  - divide data into 90% train, 10% test
- Only report treatments found in ≥ 7 samples
- Score treatments by what makes error potential worse
  - I.e. explore the worst case scenario
- Worst case scenarios:
  - Very poor developer experience and any one of
    - High resuse is a goal
    - Similar software has been used on prior missions
    - Software very simple; e.g. no intense numerical solutions.
    - Software being built by a team at one location

(so no one thinks to monitor these projects)

Stability Studies (2)

- (not reported in paper)
- Recall the SILAP constructs
  - One["complexity"] = usually 2
  - One["configManagement"] = sometimes 5;
  - One["defectTracking"] = rarely 3;
  - …
  - All["Experience"]= 0.8 to 0.9
  - All["Reuse"] = often 0.226
  - …

One: what is true about one project

All: what is true about all projects (background expert knowledge).

- Vary both the "One" and the "All" values
- What changes the conclusion first?
  - In certain cases, the Delphi "All" values
  - So, in those cases, managers could push back and say "those conclusions just come from your crazy values"
  - Action item: need to better justify those particular "All" values

Specialization Studies

Criterion	Value	Explanation
Experience	1	The developer's have built these systems before and have several years of domain experience.
Development Organization	4	Usually more than one NASA Center is involved with Human Space Flight missions.
Degree of Innovation	1	Normally, the software is not doing anything that has not been tested during a previous flight.
Use of Standards	1	Developers incorporate NASA standards as well as accepted industry standards.
Use of Configuration Management	1	Tools, as well as established methods, for configuration management are integrated into the development effort.
CMM Level	3	Methods and processes are characteristic of a Level 3 organization.
Use of Formal Reviews	1	Formal reviews are essential for the Human Space missions and they are followed and have predefined criteria.
Use of a Defect Tracking System	1	Defect tracking tools are well established at the software level and in place for the development efforts.
Use of a Risk Management System	3	Risk management tools are established at the Project level but they are not consistently used at the software level.
Artifact Maturity	1	The majority of the software artifacts are logically in a state that is similar to the schedule.

- Above:
  - all inputs picked at random
- Here:
  - pick inputs from human space flight
  - conduct a stability study on the result
- Yielded very different stable treatments
  - "Developer's experience": no longer vital
  - Rather, it is the "product complexity"

Summary

- Monte Carlo and data mining
  - Can express and explore business knowledge
- Express business knowledge in lightweight notations
  - The "elbow test"
- Stability study #1:
  - Can find stable conclusions in a large space of business possibilities
- Stability study #2:
  - Can also be used to perform V&V on the model
- Specialization study:
  - Beware general conclusions
  - Your project exists in a small corner of the space of all possible projects
  - Learn local solutions for local problems

Counter proposals

- Won't the learning just recreate the original model?
  - No: summary much smaller
  - Finds relationships that are obscure in model.
- Why not use standard Monte Carlo methods?
  - TAR3 produces much smaller theories
- Why not model with fuzzy logic, Bayes nets, decision diagrams,..?
  - All of these impose restrictions on the modeling language
  - Funnel theory: a few master variables that set the remaining "slaves:
  - Language details less important than sampling output
  - Our goal: decisions from models written any way at all
- Why not search with genetic algorithms, neural nets, …?
  - Wasted time.
  - If master variables , master variables will be obvious
- Why not search for master variables with an ATMS?
  - ATMS' complete search takes exponential time;
  - TAR3's stochastic search takes time linear on data set size

Questions?
Comments?

http://promise.unbox.org/Promise2006