Tim Menzies, WVU, USA, tim@menzies.us

LEARNING LOCAL LESSONS

Today's talk

- Claim:
  - Current SE empirical practice asks for conclusions that are are external valid
    - apply to more than one domain
  - So far, such external valid conclusions are illusive
    - Despite decades of research.
- Implications:
  - The goal is wrong
  - Seek not for general theories
    - Only for local lessons but local lessons.
- "W"
  - a baseline tool for generating local lessons

2

Two definitions of "model"

- A hypothetical description of a complex entity or process.
  - Model as output from research machine
  - The "product" of research
- A plan to create, according to a model or models
  - Model of the research machine
  - The "generator" of products
- "W" is a general model generator.

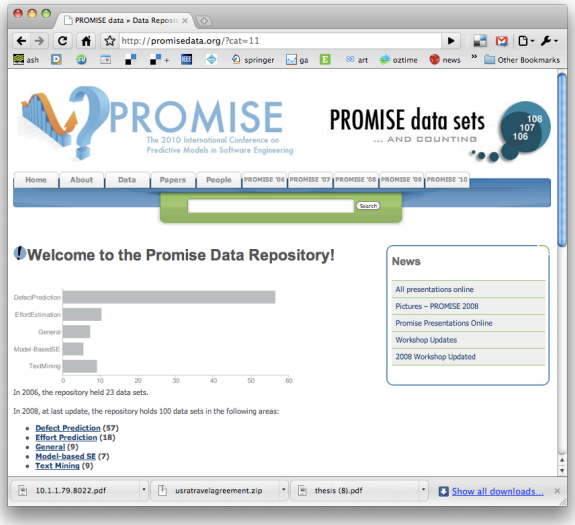
3

Disagree with me?

- Want to find some general conclusions on SE?
- Need to go somewhere to get a lot of data from different projects?

4

http://promisedata.org/data



Repository + annual conference. See you there?

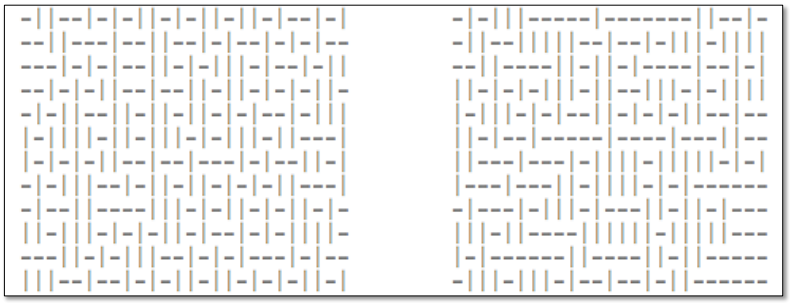
5

A WARM UP EXERCISE

6

Exercise #1

- One these these things is not like the other
  - One was generating by selecting "-" or "|" at random, 300 times.
- Which one?



7

Exercise #2

- A little experiment
- Rules
  - No one talks for the next 4 minutes
  - If you know what is about to happen, see (1)
- This will is a selective attention test
  - Count the number of times the team with the white shirt passes the ball.
  - http://www.youtube.com/v/vJG698U2Mvo&hl=en_US&fs=1&rel=0

8

What have we learned?

- Lesson #1:
  - Algorithms can be pretty dumb
  - If they don't focus on X, they see any Y, at random.
- Lesson #2:
  - Humans can be pretty dumb
  - If they mono-focus on X, you can miss Y
- Maybe, any induction process is a guess
  - And while guessing can be useful
  - Guesses can also be wrong
- Lets us a create community of agents, each with novel insights and limitations
  - Data miners working with humans
  - Maybe in combination, we can see more that separately

Wikipedia:
List of cognitive biases
• 38 decision making biases
• 30 biases in probability
• 18 social biases,
• 10 memory biases

9

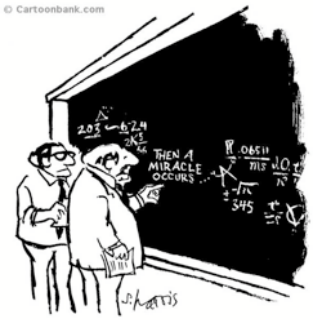
THE (VERY WEAK) STATE OF THE ART IN EMPIRICAL SE

10

Standard practice, empirical SE

Easterboork et al. (2007)

- 9 pages: selecting methods
- 3 pages: research questions
- 2 pages: empirical validity
- 2 pages: different forms of "empirical truth"
- 1 page: role of theory building
- 1 page: conclusions
- 1 page: data collection techniques
- 0 pages: data analysis
  - and then a miracle happens



- Data analysis needs more than 0 pages
  - Properly done, data analysis replaces, not augments, standard empirical methods

11

More on standard practice

- In Basili's *Goal/Question/Metric* (GQM), data collection is designed as follows:
  - *Conceptual level (goal)*: Defined w.r.t. models of quality, from various points of view and relative to a particular environment.
  - *Operational level (question)*: Define questions and models to focus on objects that characterize the assessment of that goal.
  - *Quantitative level (metric)*: Define metrics, based on models, for every question in order to collect answers in a measurable way.
- GQM is an example of "the positivist" tradition.
  - Problem statement
  - State research objective, context
  - Experiment (goals, materials, tasks, hypotheses, design)
  - Collection, hypothesis testing
  - Etc
- Release the Ph.D. students
  - Wait N years
  - Do it once, then celebrate

12

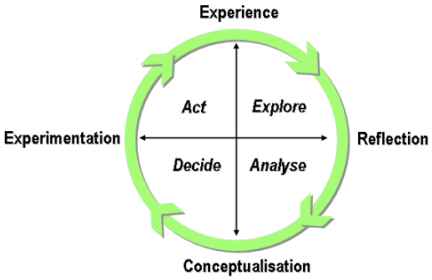
Time to change standard practice?

- Does the pace of change in modern software engineering make GQM impractical?
  - Researchers need rapid adaptation methods to keep up with this faster pace. Otherwise…
- Basili's SEL's learning organization experiment lasted ten years (from 1984 to 1994) during a period of relative stability within the NASA organization.
- Starting in 1995, the pace of change within NASA increased dramatically.
  - New projects were often outsourced
  - SEL became less the driver and more the observer, less proactive and more reactive.
  - Each project could adopt its own structure.
  - SEL-style experimentation became difficult: no longer a central model to build on.
- NASA also tried some pretty radical development methods
  - 'Faster, Better, Cheaper" lead led to certain high profile errors which were attributed to production haste, poor communications, and mistakes in engineering management.
  - When 'Faster, Better, Cheaper' ended, NASA changed, again, their development practices.
- This constant pace of change proved fatal to the SEL.
  - Basili el al. [24] describe the period 1995-2001 as one of 'retrenchment' at the SEL.

13

Done properly, data analysis replaces, not augments, standard empirical methods

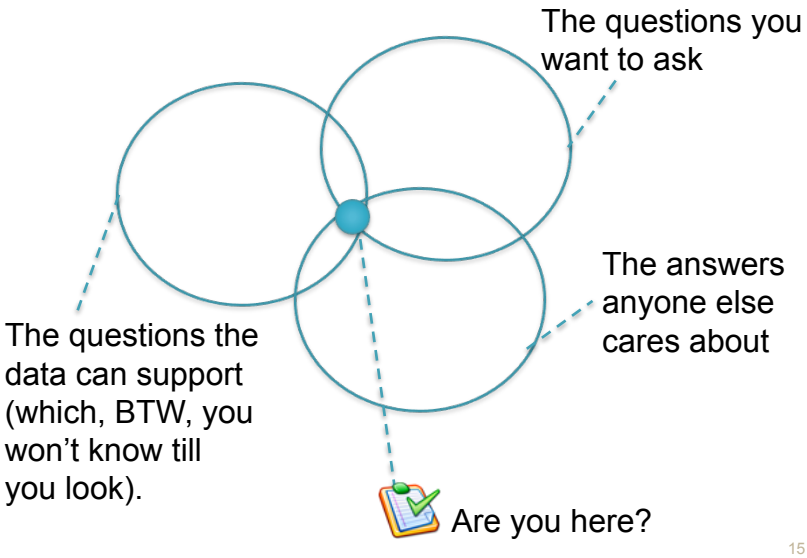
- "The only interesting answers are those which destroy the questions. -- Susan Sontag



- You can start with whatever questions you like
  - But the data may only hold answers to other questions.
- So starting with questions (version 1.0)
- Study data. If quirks, then chase the quirks
  - Now explore questions (version 2.0)

14

What to explore?



15

What have we learned from standard practice (1)?

- We spend an awful lot of time debating things that don't matter:
  - Objects,
  - aspects,
  - types,
  - etc etc

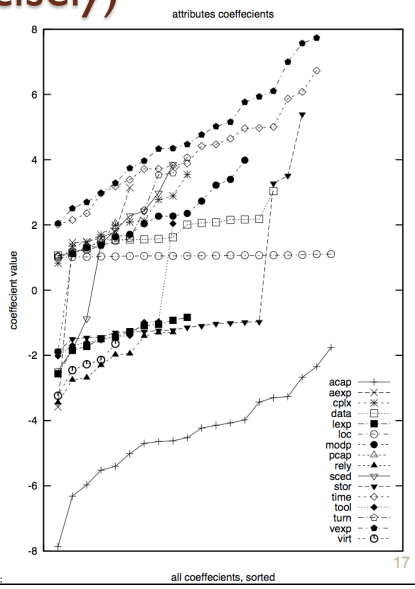
id	features	relative weight
1	Personnel/team capability	3.53
2	Product complexity	2.38
3	Time constraint	1.63
4	Required software reliability	1.54
5	Multi-site development	1.53
6	Doc. match to life cycle	1.52
7	Personnel continuity	1.51
8	Applications experience	1.51
9	Use of software tools	1.50
10	Platform volatility	1.49
11	Storage constraint	1.46
12	Process maturity	1.43
13	Language & tools experience	1.43
14	Required dev. schedule	1.43
15	Data base size	1.42
16	Platform experience	1.40
17	Arch. & risk resolution	1.39
18	Precedentedness	1.33
19	Developed for reuse	1.31
20	Team cohesion	1.29
21	Development mode	1.32
22	Development flexibility	1.26

Source: Boehm 2000. Regression results from 161 projects.

16

(But don't take treat those numbers too precisely)

- 20 experiments, using 66% of the data (selected at random)
- Linear regression:
  - Effort = b_0 + sum of $b_i * x_i$
  - Followed by a greedy back-select to prune dull variables
- Results
  - LOC influence stable
  - Some variables pruned away half the time
  - Large ranges (max – min)
  - Nine attributes even change the sign on their coefficients



17

What have we learned from standard practice (2)?

- Consider two "methods":
  - software tools, process changes, whatever,
  - Anything that we might do to "improve" (in some way) a project.
- For a list of such methods, see
  - IEEE-1012 : the V&V
  - "Handbook on software and systems engineering": Endres and Rombach, 2003
- Consider any pair of method1/ method2
  - Any studies that comparatively assess them?

18

NASA IV&V data

- From http://menzies.us/pdf/07ivv.pdf
- Step1: Find the phase that saw the most high severity errors, with least IV&V cost

concept	713	:18%
requirements	2,688	:67%
design	286	:7%
implement	182	:4%
test	146	:4%

- In that phase do the least effort thing first
  - frequency of doing "it" times the number of times you did "it"
  - If no data, write "?"
- An "heroic" study
  - Lots of business users doing lots of joins on databases never designed for inter-operative
  - Lots of "engineering judgment" on how to align terminology
- Not a reproducible study:
  - Just the best I have yet to offer
  - Has anyone else done better?
    - Nope

Less is best

phase	wbs	factor	
concept	2.1	Reuse Analysis*	109
	2.2	Software Architecture Assessment	63
	2.3	System Requirements Review	90
	2.4	Concept Document Evaluation	11
	2.5	SW/User Requirements Allocation Analysis	2
	2.6	Traceability Analysis	5
requirements	3.1	Traceability Analysis - Requirements	131
	3.2	Software Requirements Evaluation	210
	3.3	Interface Analysis - Requirements	81
	3.4	System Test Plan Analysis	4
	3.5	Acceptance Test Plan Analysis	85
	3.6	Timing and Sizing Analysis	?
design	4.1	Traceability Analysis - Design	220
	4.2	Software Design Evaluation	134
	4.3	Interface Analysis - Design	23
	4.4	Software FQT Plan Analysis	250
	4.5	Software Integration Test Plan Analysis	7
	4.6	Database Analysis	22
	4.7	Component Test Plan Analysis	?
implementation	5.1	Traceability Analysis - Code	283
	5.2	Source Code and Documentation Evaluation	545
	5.3	Interface Analysis - Code	268
	5.4	System Test Case Analysis	30
	5.5	Software FQT Case Analysis	61
	5.6	SW Integration Test Case Analysis	?
	5.7	Acceptance Test Case Analysis	?
	5.8	SW Integration Test Procedure Analysis	?
	5.9	SW Integration Test Results Analysis	?
	5.10	Component Test Case Analysis	?
	5.11	System Test Procedure Analysis	202
	5.12	Software FQT Procedure Analysis	?
test	6.1	Traceability Analysis - Test	63
	6.2	Regression Test Analysis	?
	6.3	Simulation Analysis	54
	6.4	System Test Results Analysis	202
	6.5	Software FQT Results Analysis	?
other	7.1	Operating Procedure Evaluation	?
	7.2	Anomaly Evaluation	?
	7.3	Migration Assessment	?
	7.4	Retirement Assessment	?

19

What have we learned from standard practice (3)?

- PROMISE 2005 … 2009 :
  - 64 presentations
- 48 papers
  - tried a new analysis on old data (repeatability is a GOOD thing)
  - Or reported a new method that worked once for one project.
- 4 papers
  - argued against model generality
- 9 papers
  - found issues that challenged the validity of prior results (re-assessment is a GOOD thing)
  - E.g. Menzies et al. Promise 2006
    - The variance study described above

20

What have we learned from standard practice (3, cont.)?

Only a small minority of PROMISE papers (11/64) discuss results that repeated in data sets from multiple projects

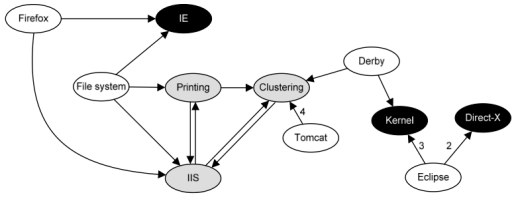
E.g. Ostrand, Weyuker, Bell '08, '09
- Same functional form
- Predicts defects for generations of AT&T software

E.g. Turhan, Menzies, Bener '08, '09
- 10 projects
  - Learn on 9
  - Apply to the 10th
- Defect models learned from NASA projects work for Turkish whitegoods software
  - Caveat: need to filter irrelevant training examples

21

What have we learned from standard practice (4)?

- The usual conclusion is that we learn that we can learn very little
- FSE'09: Zimmerman et al.
  - Defect models not generalizable
    - Learn "there", apply "here" only works in 4% of their 600+ experiments
  - Opposite to Turhan'09 resul
    - ?add relevancy filter



- ASE'09: Green, Menzies et al
  - AI search for better software project options
  - Conclusions highly dependent on local business value proposition
- And others
  - TSE '01, '05: Shepperd et al
    - Any conclusion regarding "best" effort estimator varies by data sets, performance criteria, random selection train/test set
  - TSE'06: Menzies, Greenwald:
    - attributes selected by FSS vary wildly across projects
  - Zannier et al ICSE'06:
    - picked 5% (at random) ICSE claiming to be "empirical,"
    - very few of them (2%) compare methods from multiple researchers

22

The gods are angry

- Fenton at PROMISE' 07
  - "... much of the current software metrics research is inherently irrelevant to the industrial mix ..."
  - "... any software metrics program that depends on some extensive metrics collection is doomed to failure ..."
- Budgen & Kitchenham:
  - "Is Evidence Based Software Engineering mature enough for Practice & Policy? "
  - Need for better reporting: more reviews.
  - Empirical SE results too immature for making policy.
- Basili : still far to go
  - But we should celebrate the progress made over the last 30 years.
  - And we are turning the corner

23

A new hope (actually, quite old)

- Experience factories
  - Method for find local lessons
- Basili'09 (pers. comm.):
  - "All my papers have the same form.
  - "For the project being studied, we find that changing X improved Y."
- Translation (mine):
  - Even if we can't find general models (which seem to be quite rare)….
  - … we can still research general methods for finding local lessons learned

24

If we can't find general models, is it science?

Popper '60: Everything is a "hypothesis"
- And the good ones have weathered the most attack
- SE "theories" aren't even "hypotheses"

Endres & Rombach '03: Distinguish "observations", "laws", "theory"
- Laws predict repeatable observations
- Theories explain laws
- Laws are either hypotheses (tentatively accepted) or conjectures (guesses)

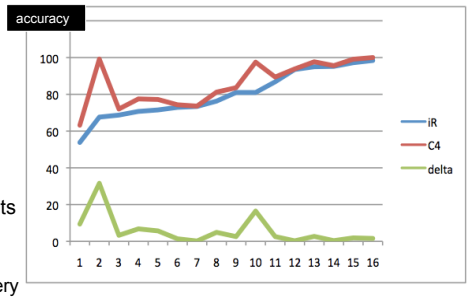
Gregor'06 : 5 types of "theory":
1. Analysis (e.g. ontologies, taxonomies)
2. Explanation (but it is hard to explain "explanation")
3. Prediction (some predictors do not explain)
4. Explanation and prediction
5. "models" for design + action
   - Don't have to be "right"
   - Just "useful"
   - A.k.a. Endres & Rombach's "laws"?

25

The rest of this talk: "W" (a "local lessons" finder)

26/46

- Bayesian case-based contrast-set learner
  - uses greedy search
  - illustrates the "local lessons" effect
  - offers functionality missing in the effort-estimation literature
- Fast generator of baseline results
  - There are too few baseline results
  - And baseline results can be very interesting (humbling).
- A very (very) simple algorithm
  - Should add it to your toolkit
  - At least, as the "one to beat"



Holte'93
- C4: builds decision trees "N" deep
- 1R: builds decision trees "1" deep
- For datasets with 2 classes, 1R ≈ C4

26

CONTRAST SET LEARNING WITH "W"

27

"W"= Simple (Bayesian) Contrast Set Learning (in linear time)

Mozini: KDD '04

- "best" = target class (e.g. "survive")
- "rest" = other classes
- x = any range (e.g. "sex=female")
- f(x|c) = frequency of x in class c

- b = f(x | best) / F(best)
- r = f(x | rest) / F(rest)

- LOR= log(odds ratio) = log(b/r)
  - ? normalize 0 to max = 1 to 100

- s = sum of LORs
  - e = 2.7183 …
  - p = F(B) / (F(B) + F(R))
  - P(B) = 1 / (1 + e^(-1*ln(p/(1 - p)) - s))

28

"W"= Simple (Bayesian) Contrast Set Learning (in linear time)

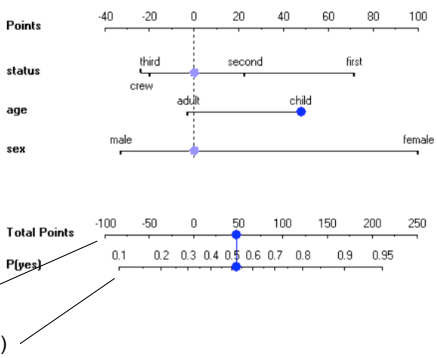
Mozini: KDD '04

- "best" = target class (e.g. "survive")
- "rest" = other classes
- x = any range (e.g. "sex=female")
- f(x|c) = frequency of x in class c

- b = f(x | best) / F(best)
- r = f(x | rest) / F(rest)

- LOR= log(odds ratio) = log(b/r)
  - ? normalize 0 to max = 1 to 100

- s = sum of LORs
  - e = 2.7183 …
  - p = F(B) / (F(B) + F(R))
  - P(B) = 1 / (1 + e^(-1*ln(p/(1 - p)) - s))



29

"W"= Simple (Bayesian) Contrast Set Learning (in linear time)

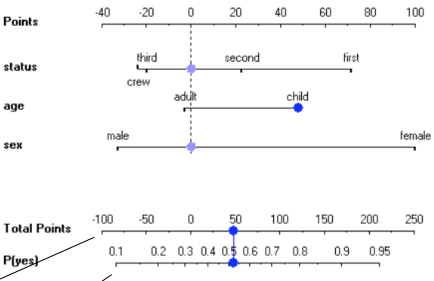
Mozini: KDD '04

- "best" = target class (e.g. "survive")
- "rest" = other classes
- x = any range (e.g. "sex=female")
- f(x|c) = frequency of x in class c

- b = f(x | best) / F(best)
- r = f(x | rest) / F(rest)

- LOR= log(odds ratio) = log(b/r)
  - ? normalize 0 to max = 1 to 100

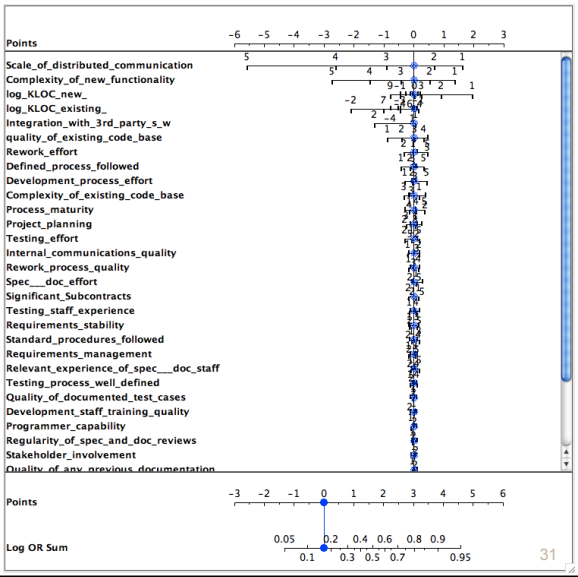
- s = sum of LORs
  - e = 2.7183 …
  - p = F(B) / (F(B) + F(R))
  - P(B) = 1 / (1 + e^(-1*ln(p/(1 - p)) - s))



"W":
1) Discretize data and outcomes
2) Count frequencies of ranges in classes
3) Sort ranges by LOR
4) Greedy search on top ranked ranges

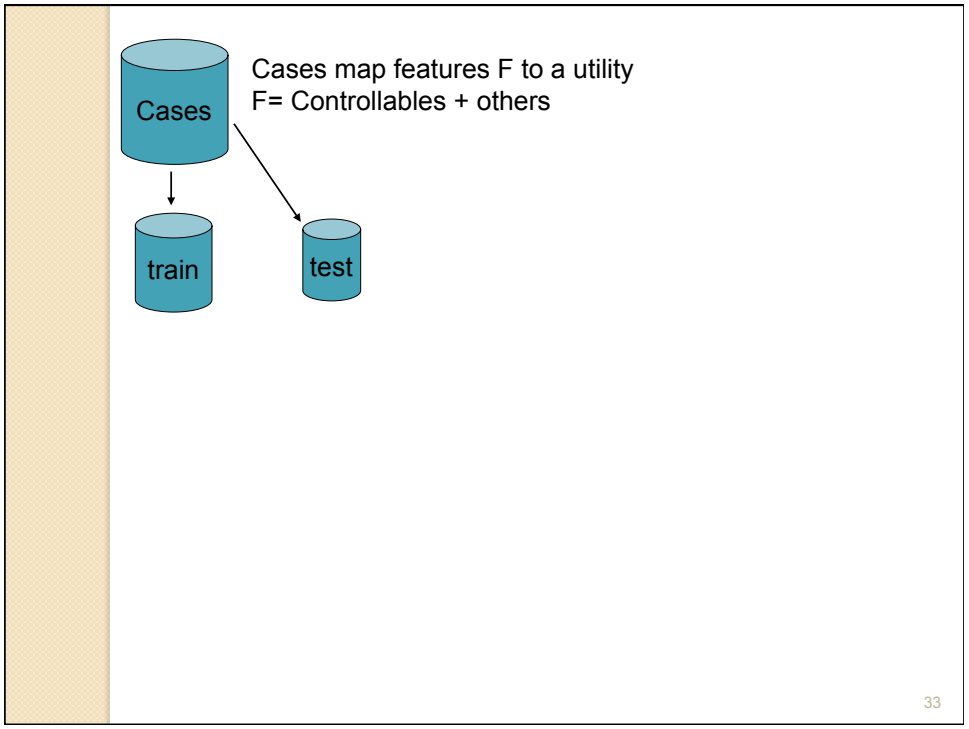
LOR and Defect Prediction

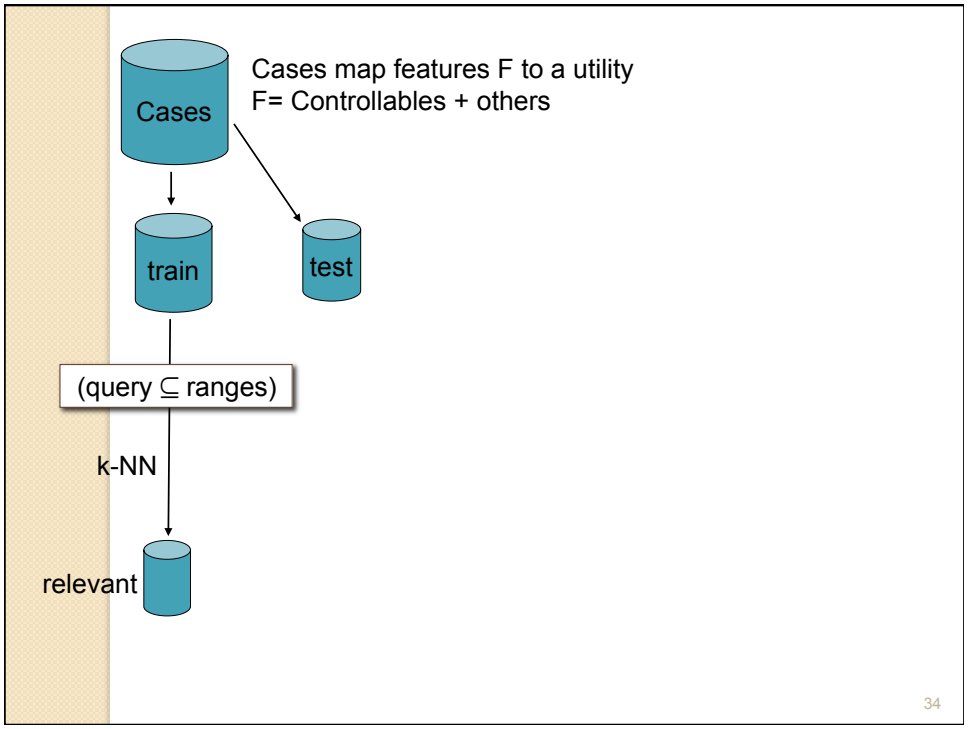
- Data from Norman Fenton's Bayes Net
- Classes=
  - round(log2(defects/KLOC))/2)
- Target class = max(class)
  - I.e. worse defects
- Only a few features matter
- Only a few ranges of those features matter

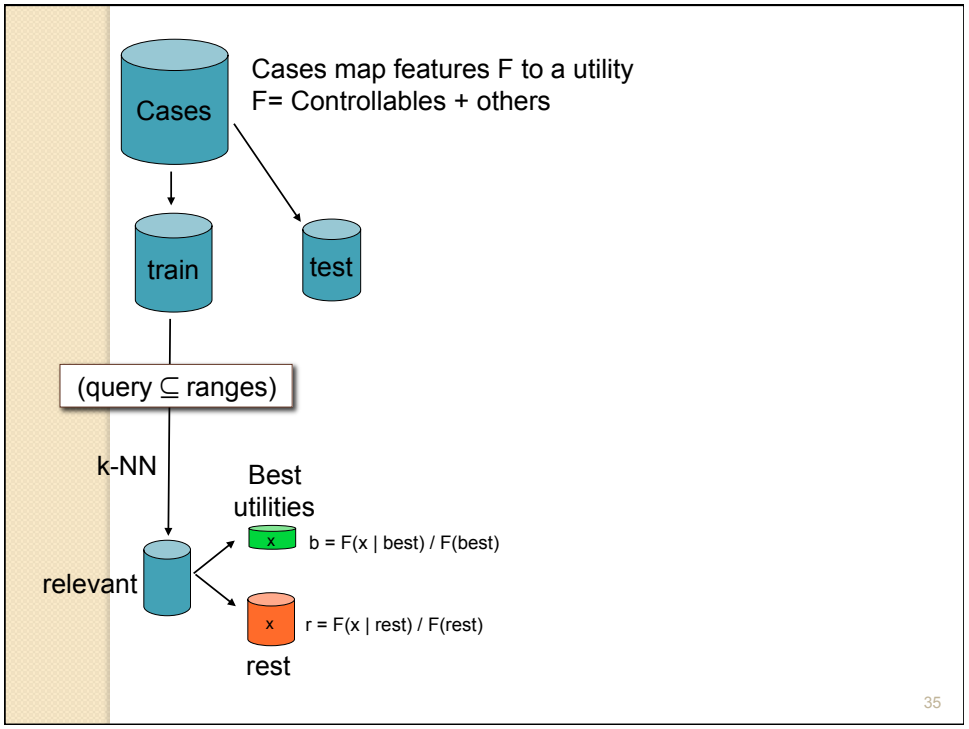


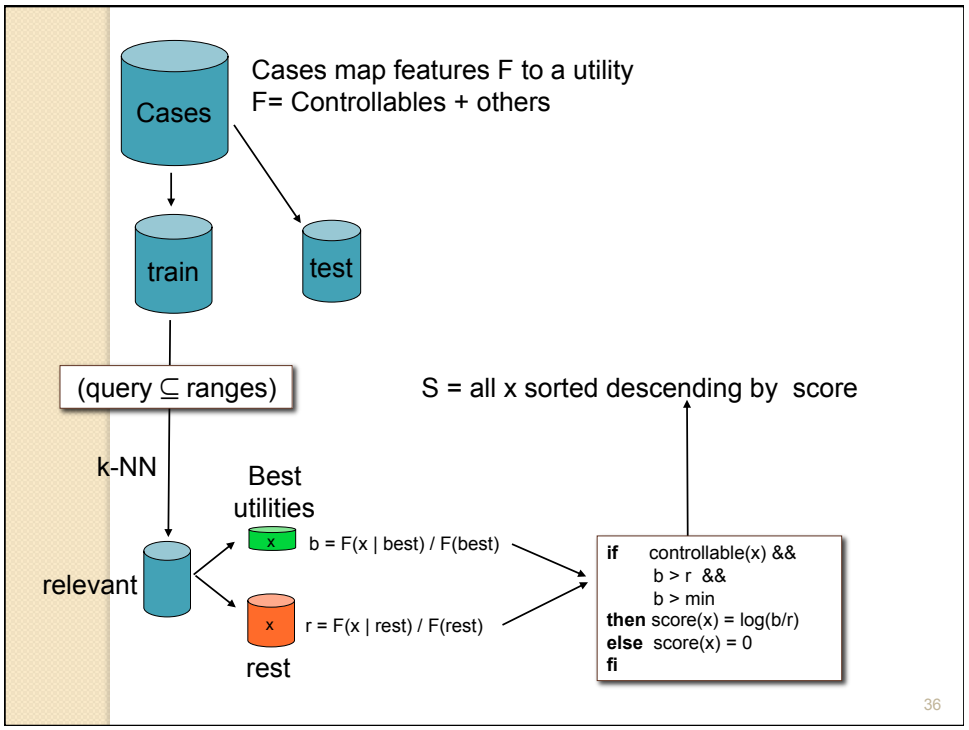
"W" + CBR: Preliminaries

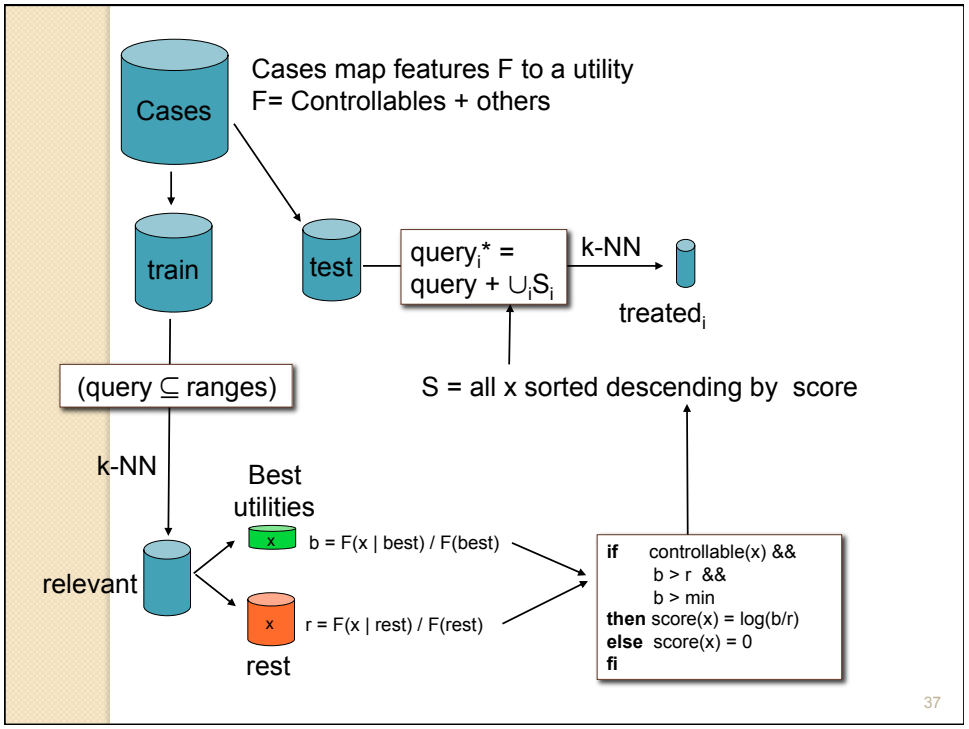
- "Query"
  - What kind of project you want to analyze; e.g.
    - Analysts not so clever,
    - High reliability system
    - Small KLOC
- "Cases"
  - Historical records, with their development effort
- Output:
  - A recommendation on how to change our projects in order to reduce development effort

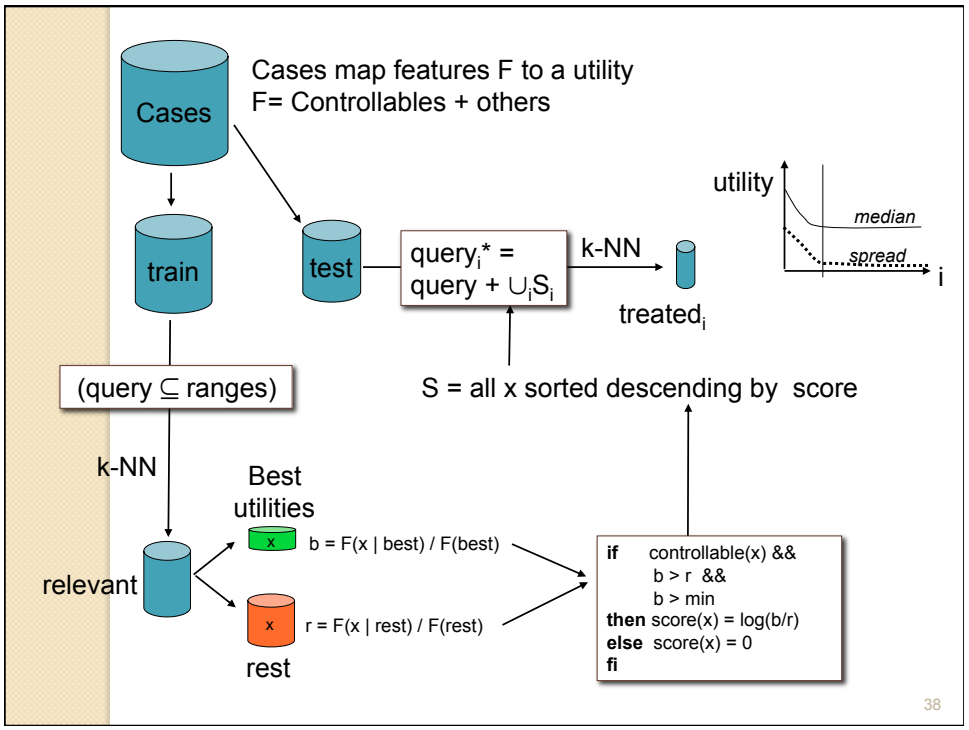

Cases
Cases map features F to a utility
F= Controllables + others
train
test
33

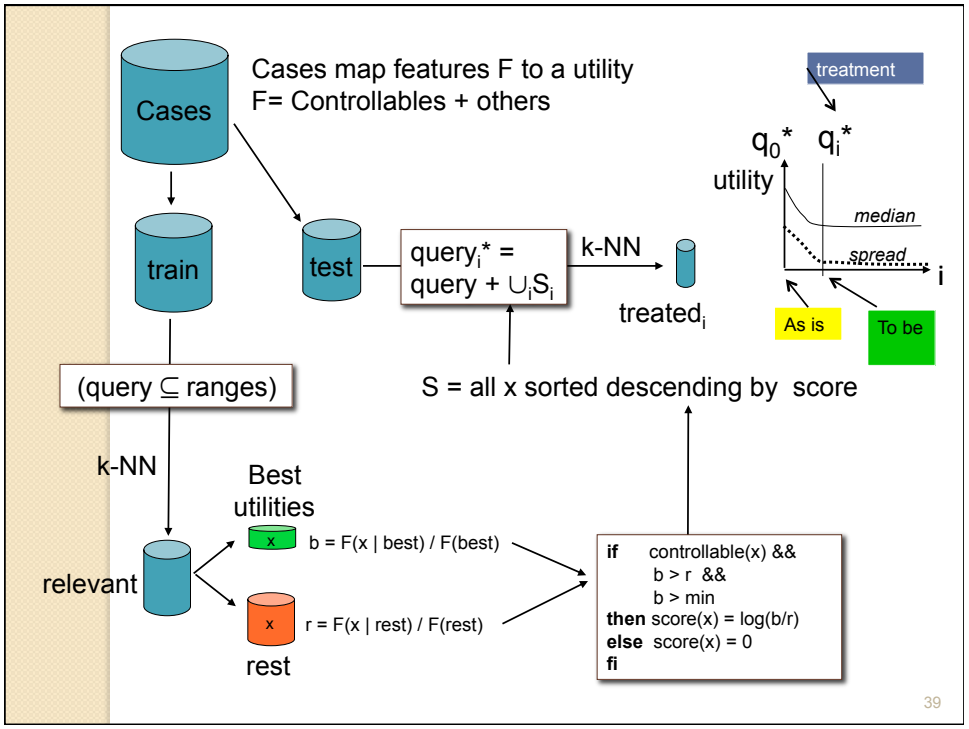

Cases
Cases map features F to a utility
F= Controllables + others
train
test
(query ⊆ ranges)
k-NN
relevant
34


Cases
Cases map features F to a utility
F= Controllables + others
train
test
(query ⊆ ranges)
k-NN
Best
utilities
relevant
x
b = F(x | best) / F(best)
x
r = F(x | rest) / F(rest)
rest
35


Cases
Cases map features F to a utility
F= Controllables + others
train
test
(query ⊆ ranges)
S = all x sorted descending by score
k-NN
Best
utilities
relevant
x
b = F(x | best) / F(best)
x
r = F(x | rest) / F(rest)
rest
if controllable(x) &&
b > r &&
b > min
then score(x) = log(b/r)
else score(x) = 0
fi
36


Cases
Cases map features F to a utility
F= Controllables + others
train
test
query$_i$* =
query + $\cup_i S_i$
k-NN
treated$_i$
(query ⊆ ranges)
S = all x sorted descending by score
k-NN
Best
utilities
relevant
x
b = F(x | best) / F(best)
x
r = F(x | rest) / F(rest)
rest
if controllable(x) &&
b > r &&
b > min
then score(x) = log(b/r)
else score(x) = 0
fi
37


Cases
Cases map features F to a utility
F= Controllables + others
utility
median
spread
i
train
test
query$_i$* =
query + $\cup_i S_i$
k-NN
treated$_i$
(query ⊆ ranges)
S = all x sorted descending by score
k-NN
Best
utilities
relevant
x
b = F(x | best) / F(best)
x
r = F(x | rest) / F(rest)
rest
if controllable(x) &&
b > r &&
b > min
then score(x) = log(b/r)
else score(x) = 0
fi
38

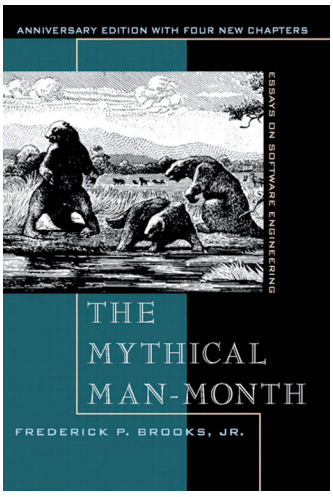

Cases
Cases map features F to a utility
F= Controllables + others
treatment
q_0* q_i*
utility
median
spread
i
train
test
query$_i$* =
query + $\cup_i S_i$
k-NN
treated$_i$
As is
To be
(query ⊆ ranges)
S = all x sorted descending by score
k-NN
Best
utilities
relevant
x
b = F(x | best) / F(best)
x
r = F(x | rest) / F(rest)
rest
if controllable(x) &&
b > r &&
b > min
then score(x) = log(b/r)
else score(x) = 0
fi
39


#1: Brooks's Law

Some tasks have inherent temporal constraints

41

Brooks's Law (1975)



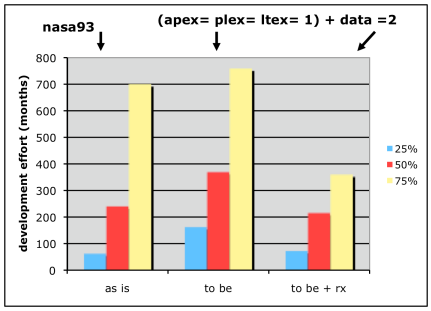
"Adding manpower (sic) to a late project makes it later".

Inexperience of new comers
- Extra communication overhead
- Slower progress

42

"W", CBR, & Brooks's law

- Can we mitigate for decreased experience?
- Data:
  - Nasa93.arff
    (from promisedata.org)
- Query:
  - Applications Experience
    - "aexp=1" : under 2 months
  - Platform Experience
    - "plex=1" : under 2 months
  - Language and tool experience
    - "ltex = 1" : under 2 months



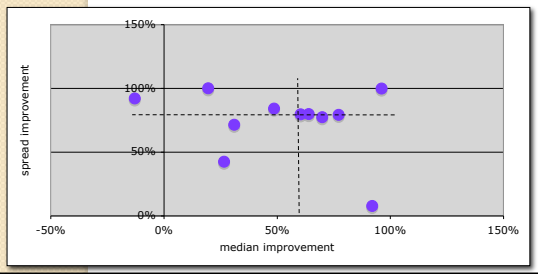
- For nasa93, inexperience does not always delay the project
  - if you can reign in the DB requirements.
- So generalities may be false
  - in specific circumstances

Need ways to quickly build and maintain domain-specific SE models

EG #2 , #3,…. #13

Results (distribution of development efforts in q_i*)

		X = as is		Y = to be		(X-Y) / X	
cases	query	median	spread	median	spread	median	spread
coc81	allSmall	70	920	79	73	-13%	92%
coc81	flight	87	281	70	0	20%	100%
nasa93	osp2	409	653	300	376	27%	42%
coc81	osp2	87	483	60	138	31%	71%
nasa93	osp	409	781	210	125	49%	84%
nasa93	allSmall	409	588	162	120	60%	80%
coc81	allLarge	50	158	18	32	64%	80%
nasa93	allLarge	300	660	90	150	70%	77%
nasa93	ground	360	481	82	100	77%	79%
coc81	osp	88	483	7	446	92%	8%
coc81	ground	156	478	6	1	96%	100%
nasa93	flight	360	474				



Cases from *promisedata.org/data*

Median = 50% percentile
Spread = 75% - 25% percentile

Improvement = (X - Y) / X
- X = as is
- Y = to be
- more is better

Usually:
- spread ≥ 75% improvement
- median ≥ 60% improvement

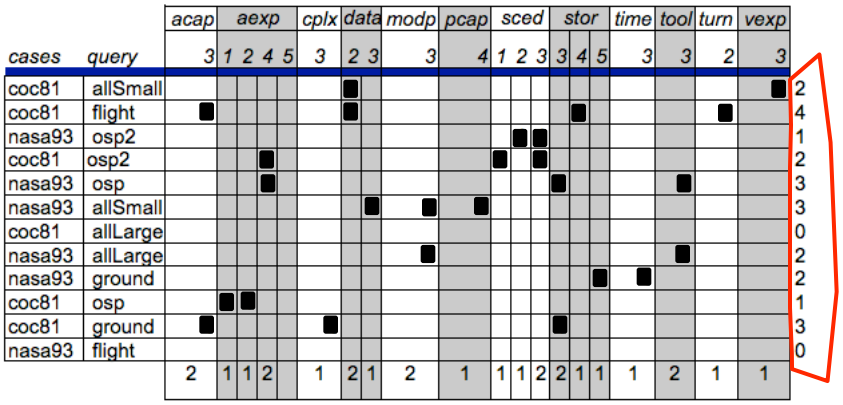
45

But that was so easy

- And that's the whole point
- Yes, finding local lessons learned need not be difficult
- Strange to say…
  - There are no references in the CBR effort estimation literature for anything else than estimate = nearest neighbors
  - No steps beyond into planning , etc
  - Even though that next steps is easy

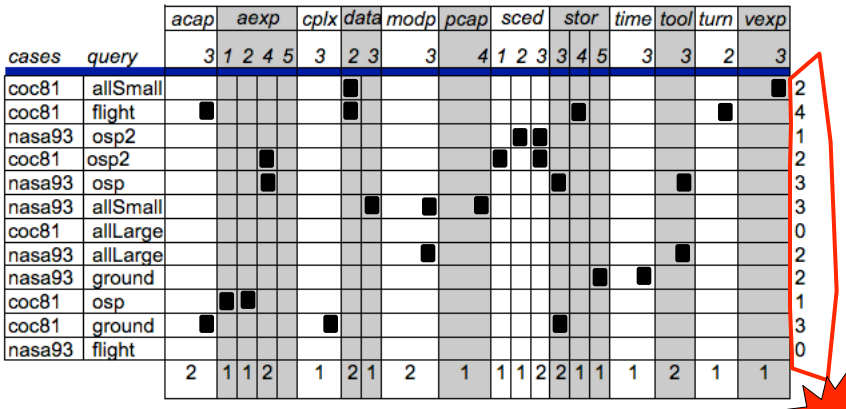
46

What should change?
$(q_j^* - q_o^*)$



		acap	aexp				cplx	data		modp	pcap	sced			stor			time	tool	turn	vexp	
cases	query	3	1	2	4	5	3	2	3	3	4	1	2	3	3	4	5	3	3	2	3	
coc81	allSmall							■													■	2
coc81	flight	■						■								■				■		4
nasa93	osp2												■	■								1
coc81	osp2				■							■		■								2
nasa93	osp				■										■				■			3
nasa93	allSmall								■	■	■											3
coc81	allLarge																					0
nasa93	allLarge									■									■			2
nasa93	ground															■		■				2
coc81	osp		■	■																		1
coc81	ground	■					■								■							3
nasa93	flight																					0
		2	1	1	2		1	2	1	2	1	1	1	2	2	1	1	1	2	1	1	

Good news:
improving estimates requires very few changes



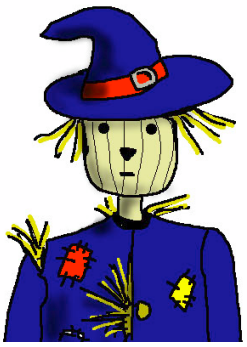
		acap	aexp				cplx	data		modp	pcap	sced			stor			time	tool	turn	vexp	
cases	query	3	1	2	4	5	3	2	3	3	4	1	2	3	3	4	5	3	3	2	3	
coc81	allSmall							■													■	2
coc81	flight	■						■								■				■		4
nasa93	osp2												■	■								1
coc81	osp2				■							■		■								2
nasa93	osp				■										■				■			3
nasa93	allSmall								■	■	■											3
coc81	allLarge																					0
nasa93	allLarge									■									■			2
nasa93	ground															■		■				2
coc81	osp		■	■																		1
coc81	ground	■					■								■							3
nasa93	flight																					0
		2	1	1	2		1	2	1	2	1	1	1	2	2	1	1	1	2	1	1	

Not-so-good news: local lessons very local

cases	query	acap	aexp				cplx	data		modp	pcap	sced			stor			time	tool	turn	vexp	
		3	1	2	4	5	3	2	3	3	4	1	2	3	3	4	5	3	3	2	3	
coc81	allSmall							■													■	2
coc81	flight	■						■								■				■		4
nasa93	osp2												■	■								1
coc81	osp2				■							■		■								2
nasa93	osp				■										■				■			3
nasa93	allSmall								■	■	■											3
coc81	allLarge																					0
nasa93	allLarge									■									■			2
nasa93	ground															■		■				2
coc81	osp		■	■																		1
coc81	ground	■					■								■							3
nasa93	flight																					0
		2	1	1	2		1	2	1	2	1	1	1	2	2	1	1	1	2	1	1	

49

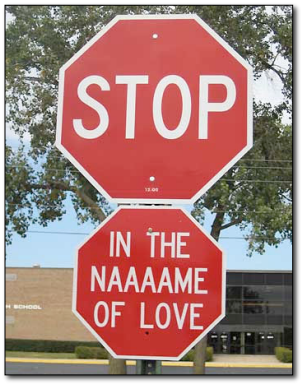
Q: Can we do better than "W"? A: Most certainly!



- "W" contains at least a dozen arbitrary design decisions
  - Which is best?
- But the algorithm is so simple
  - It should least be a baseline tool
  - Against which we compare supposedly more sophisticated methods.
  - The straw man
- Methodological advice
  - Before getting complex, get simple
  - Warning: often: my straw men don't burn

50

In conclusion …



51

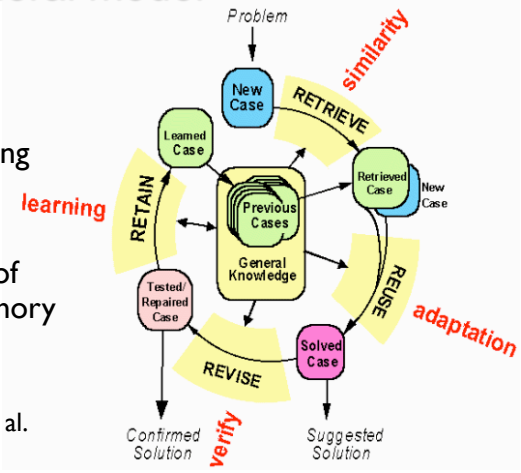
Certainly, we should always strive for generality

- But don't be alarmed if you can't find it.
- The experience to date is that,
  - with rare exceptions,
  - SE does not lead to general models
- But that's ok
  - Very few others have found general models (in SE)
  - E.g. Turhan, Menzies, Ayse'09
- Anyway
  - If there are few general results, there may be general methods to find local results
    - Seek not "models as products"
    - But general models to generate products

52

Btw, constantly (re)building local models is a general model

- Case-based reasoning
- Kolodner's theory of reconstructive memory
- The Yale group
  - Shank & Riesbeck et al.
  - Memory, not models
  - Don't "think", remember



53

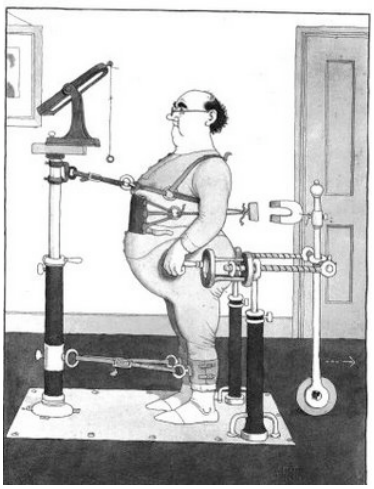
Kludges: they work

Ask some good old fashioned AI types

Minsky'86: "Society of Mind"
- The brain is a set of 1000+ kludges

Feigenbaum'78
- Don't take your heart attack to the Maths Dept.
  - Were they will diagnose and treat you using first principles
- Instead, go to the E.R room
  - Staffed by doctors who spent decades learning the quirks of drugs, organs, diseases, people, etc

Seek out those that study kludges.
- You'll be treated faster
- You'll live longer



54

Disagree with me?

- Want to find some general conclusions on SE?
- Need to go somewhere to get a lot of data from different projects?

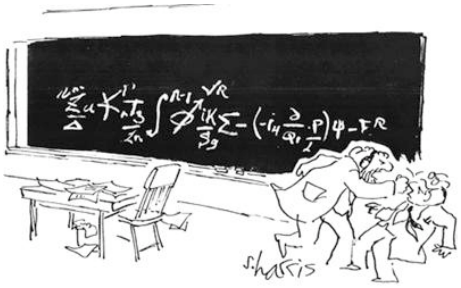
55

http://promisedata.org/data

Repository + annual conference. See you there?

56

Questions?
Comments?



"You want proof? I'll give you proof!"