



Adaptive embedded systems

Devika Subramanian
Department of Computer Science
Rice University

<http://www.cs.rice.edu/~devika>

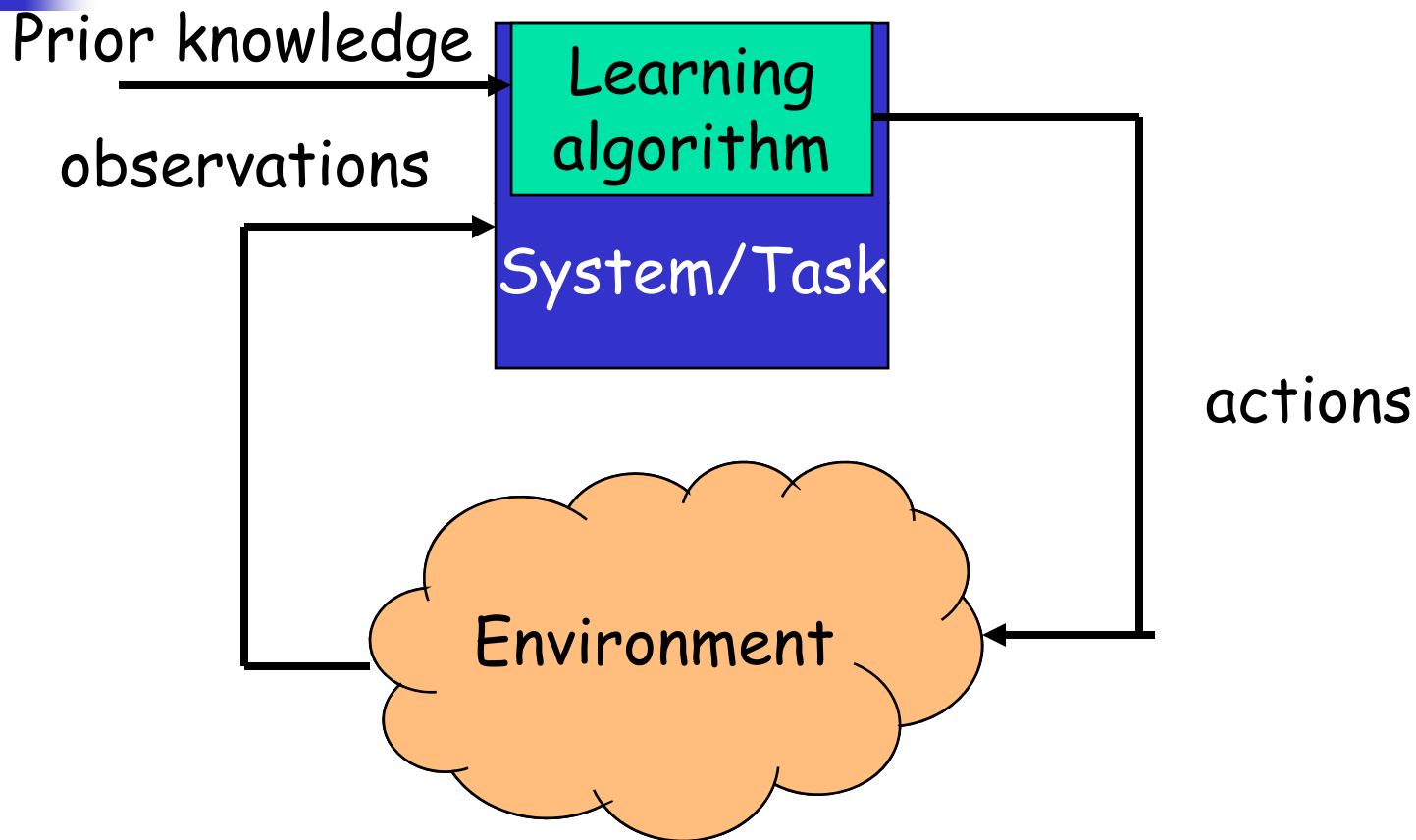




Learning



Embedded learning



Calculate decisions on the basis of **learned** models of systems



Why embed learning?

- We cannot calculate and implement an action-choice/decision-making strategy for the system at design time.
 - System dynamics are unknown/partially known.
 - System dynamics change with time.
 - A one-size-fits-all solution is not appropriate - customization is needed.



Research issues

- **Modeling:** What aspects of the task, environment and system dynamics do we need to observe and model for decision making?
- **Learning:** How can we build and maintain embedded models in changing environments?
- **Decision making/acting:** How can we use these models effectively to make decisions with scarce resources in changing environments?



Research questions

- What is the basic science of adaptive embedded systems?
 - What data should be gathered to make predictions for given task classes?
 - What kind of model should be learned (e.g., deterministic or probabilistic) for given task classes?
 - How do we evaluate learned models?
 - What algorithms should we use to learn these models from data? How can we scale them to work on large data sets in non-stationary environments?

Approach

- Design and validate algorithms for large-scale real world applications.
- Publish in the application community journals; get community to routinely use the methods.
- Abstract task-level analysis and present methods to the UAI/Machine Learning community.



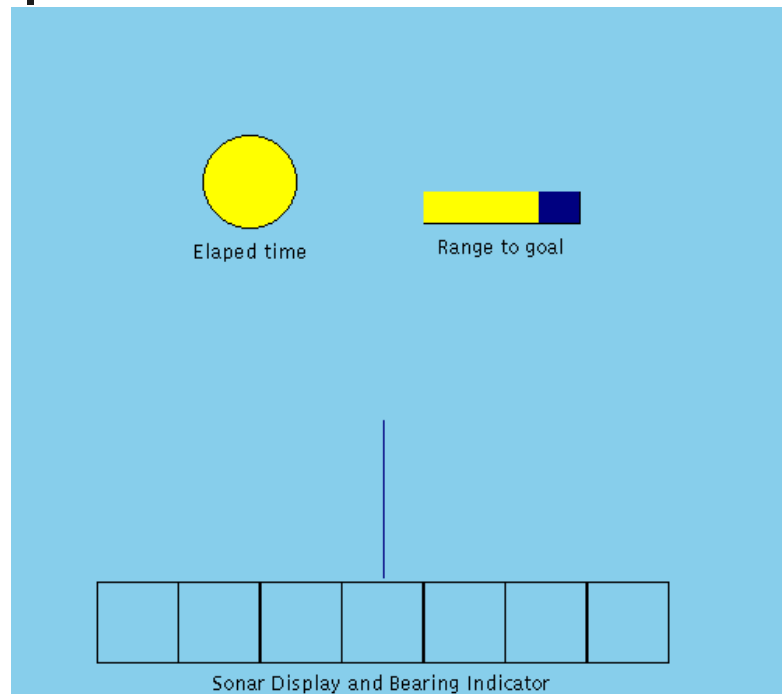


Roadmap of talk

- Four case-studies
 - Unknown system, changing dynamics
 - Tracking human learning on a complex visual-motor task.
 - Predicting the evolution of international conflict.
 - Unknown system, customization needed
 - Learning disruptions in metabolic processes in cancer cells.
 - Designing customized compiler optimization sequences for application programs.

Submarine School 101

The NRL Navigation Task



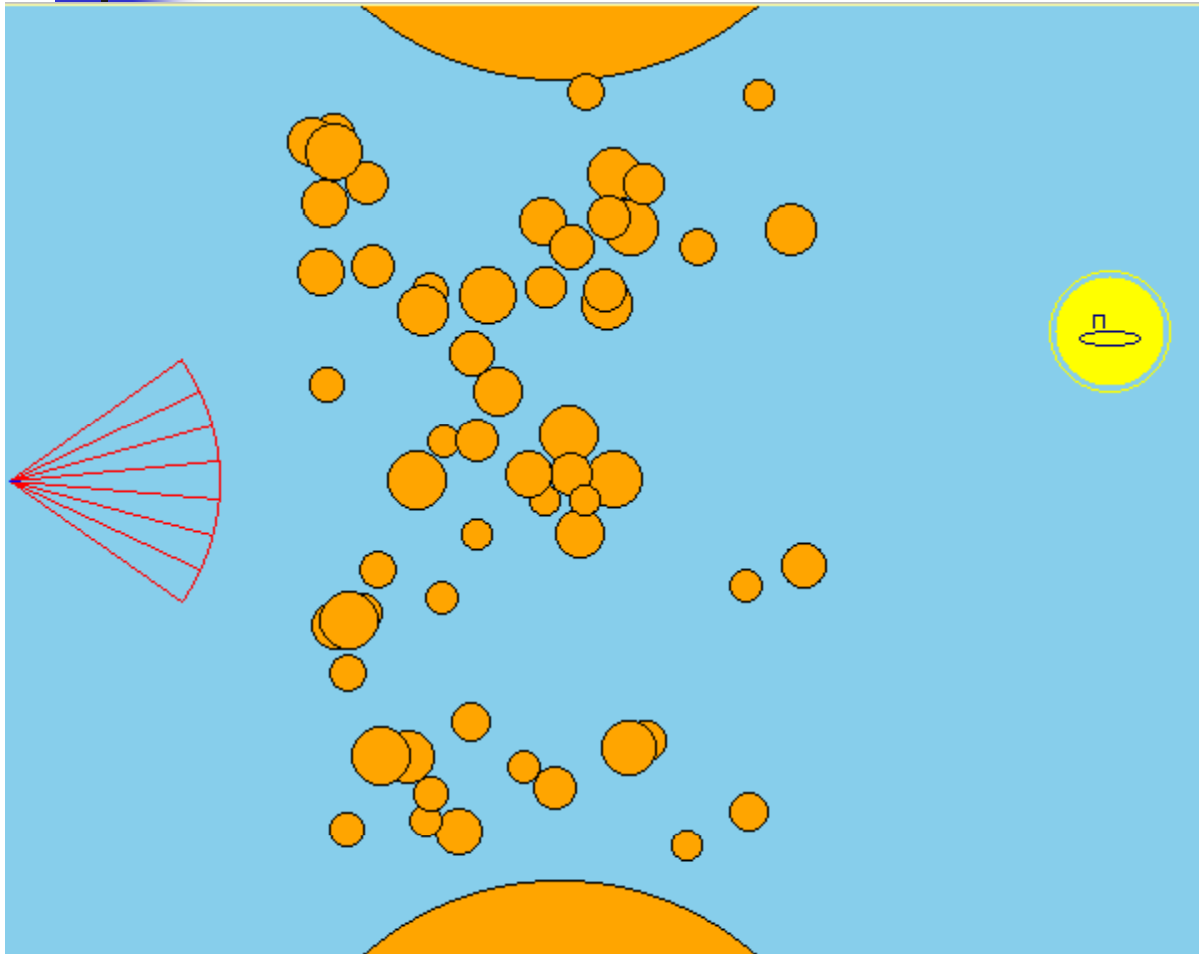
- Pilot a submarine to a goal through a minefield in a limited time period
- Distance to mines revealed via seven discrete sonars
- Time remaining, as-the-crow-flies distance to goal, and bearing to goal is given
- Actions communicated via a joystick interface



50% of class weeded out by this game!



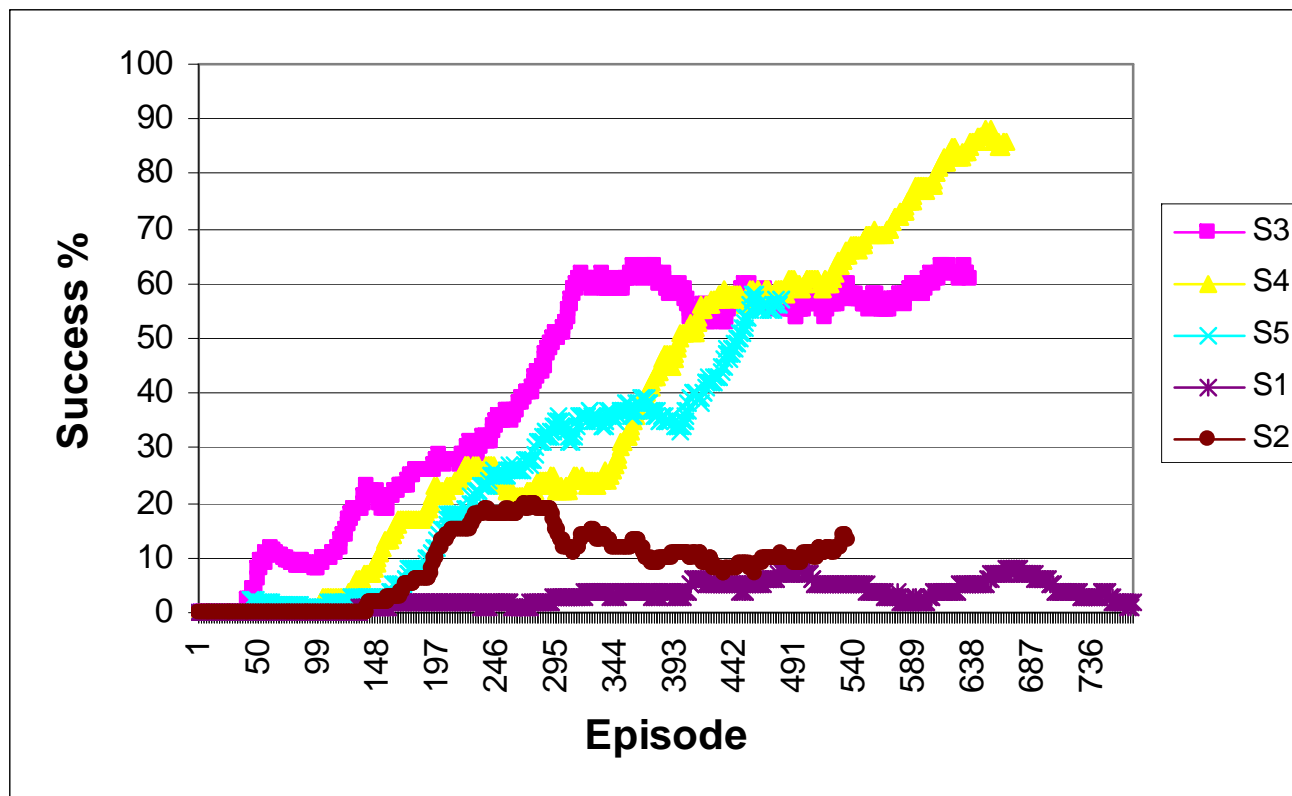
The NRL Navigation Task



Mine configuration changes with every game.

Game has a strategic and a visual-motor component!

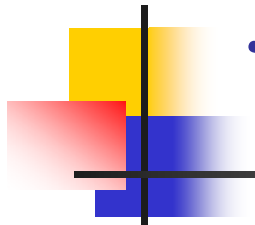
Learning curves



Successful learners look similar: plateaus between improvements

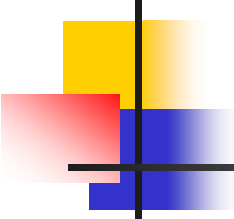
Unsuccessful learners are DOA!

Navy takes 5 days to tell if a person succeeds/fails.



Task Questions

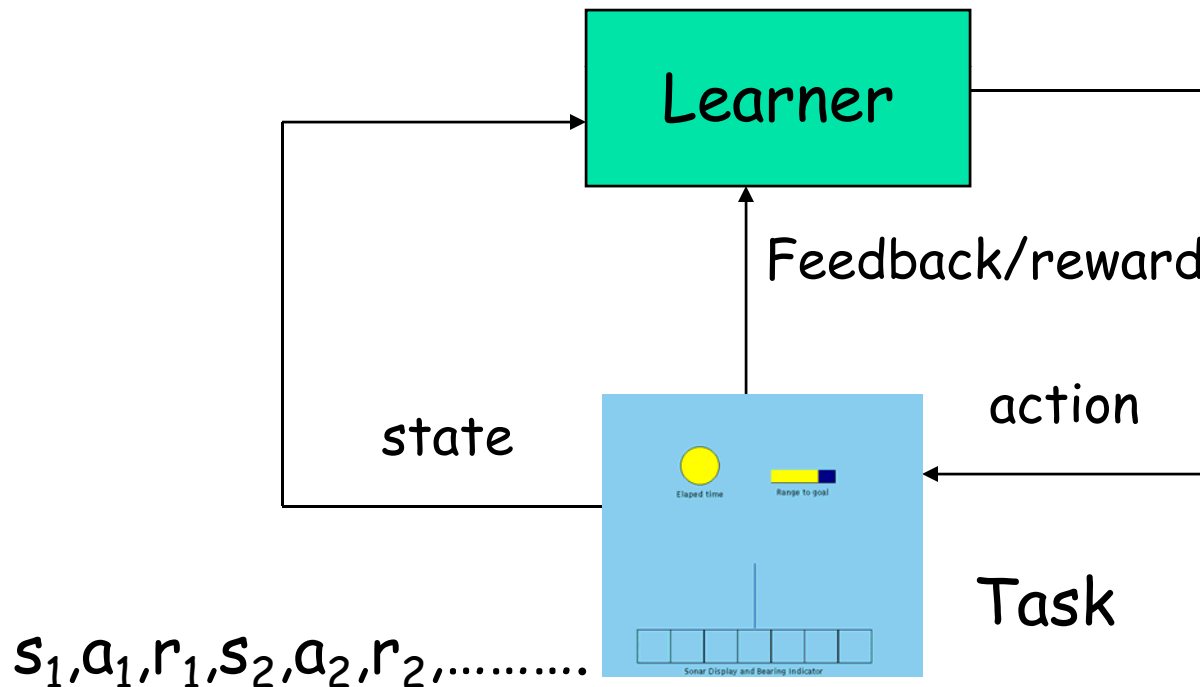
- Is the game hard? What is the source of complexity?
- Why does human performance plateau out at 80%? Is that a feature of the human learning system or the game? Can machine learners achieve higher levels of competence?
- Can we understand why humans learn/fail to learn the task? Can we detect inability to learn early enough to intervene?
- How can we actively shape human learning on this task?



Mathematical characteristics of the NRL task

- A partially observable Markov decision process which can be made fully observable by augmentation of state with previous action.
- State space of size 10^{14} , at each step a choice of 153 actions (17 turns and 9 speeds).
- Feedback at the end of up to 200 steps.
- Challenging for both humans and machines.

Reinforcement learning



Reinforcement Learning, Barto and Sutton, MIT Press, 1998.



Q-Learning Algorithm

1. Initialize $Q(s, a)$ to small random values, $\forall s, a$
 2. Observe state, s
 3. Pick an action, a , and do it
 4. Observe next state, s' , and reward, r
 5. * $Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(r + \max_{a'} Q(s', a'))$
 6. Go to 2
- $0 \leq \alpha \leq 1$ is the learning rate, decayed over time.

$Q(s,a)$ = is the sum of rewards from s to terminal state upon taking action a out of state s .

* We use eligibility traces.



Reinforcement learning/NRL task

- Representational hurdles
 - State and action spaces have to be manageably small.
 - Good intermediate feedback in the form of a non-deceptive progress function needed.
- Algorithmic hurdles
 - Appropriate credit assignment policy needed to handle the two types of failures (timeouts and explosions are different).
 - Q-learning is too slow to converge (because there are up to 200 steps in a single training episode).



State space design

- Binary distinction on sonar: is it > 50 ?
- Six equivalence classes on bearing: 12, $\{1,2\}$, $\{3,4\}$, $\{5,6,7\}$, $\{8,9\}$, $\{10,11\}$
- State space size = $2^7 * 6 = 768$.
- Discretization of actions
 - speed: 0, 20 and 40.
 - turn: -32, -16, -16, -8, 0, 8, 16, 32.



The dense reward function

$r(s,a,s')$ = 0 if s' is a state where player hits mine.
= 1 if s' is a goal state
= 0.5 if s' is a timeout state

= 0.75 if s is an all-blocked state and s' is a not-all-blocked state
= 0.5 + Diff in sum of sonars/1000 if s' is an all-blocked state
= 0.5 + Diff in range/1000 + $\text{abs}(\text{bearing} - 6)/40$ otherwise

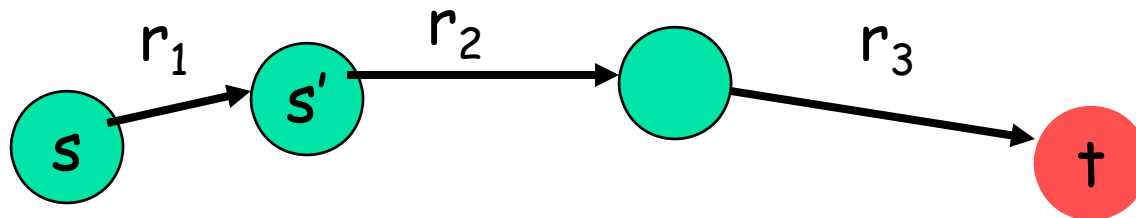


Credit assignment policy

- Penalize the last action alone in a sequence which ends in an explosion.
- Penalize all actions in sequence which ends in a timeout.

Simplification of value estimation

- Estimate the average local reward for each action in each state.



Instead of learning Q

$$Q(s, a) = \alpha[r + \max_{a'} Q(s', a')] + (1 - \alpha)Q(s, a)$$

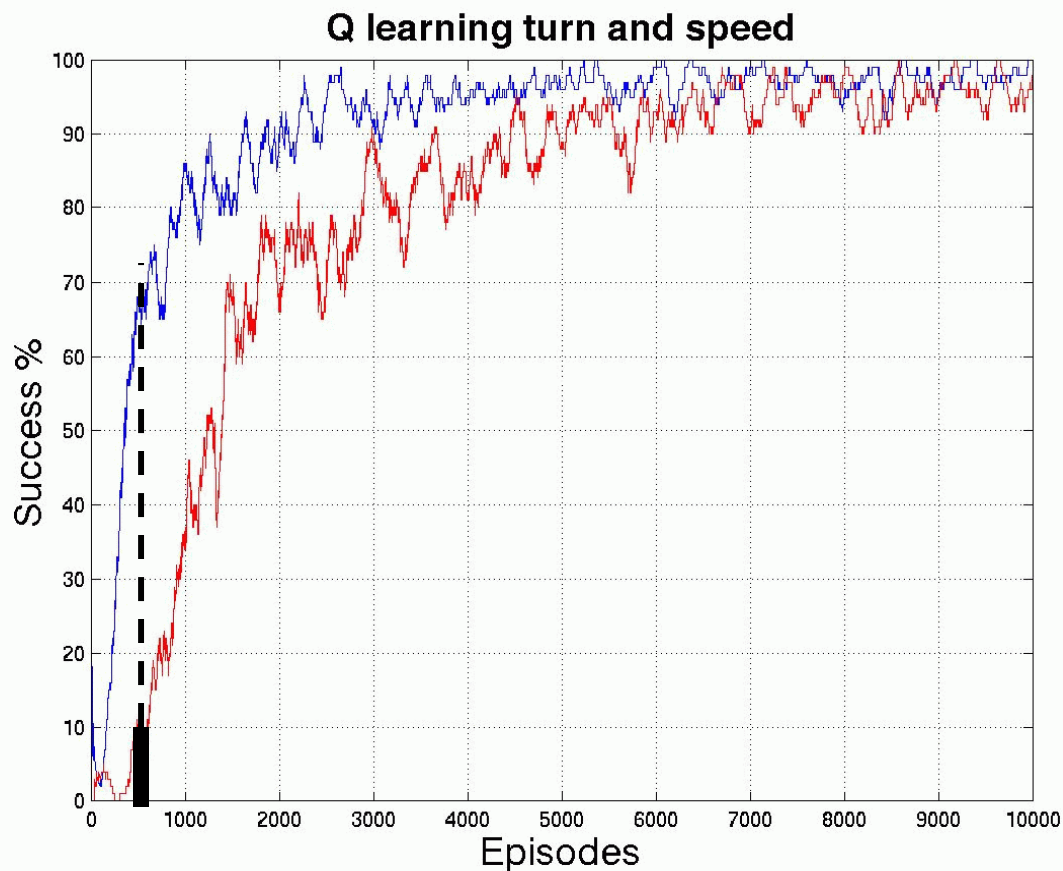
We maintain an approximation

$$Q'(s, a) = (\text{running avg of rewards at } s \text{ for } a) * \text{pct of wins from } s$$

$Q(s, a)$ = is the sum of rewards from s to terminal state.

Open question:
When does this approx work?

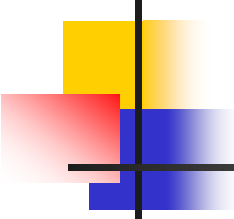
Results of learning complete policy



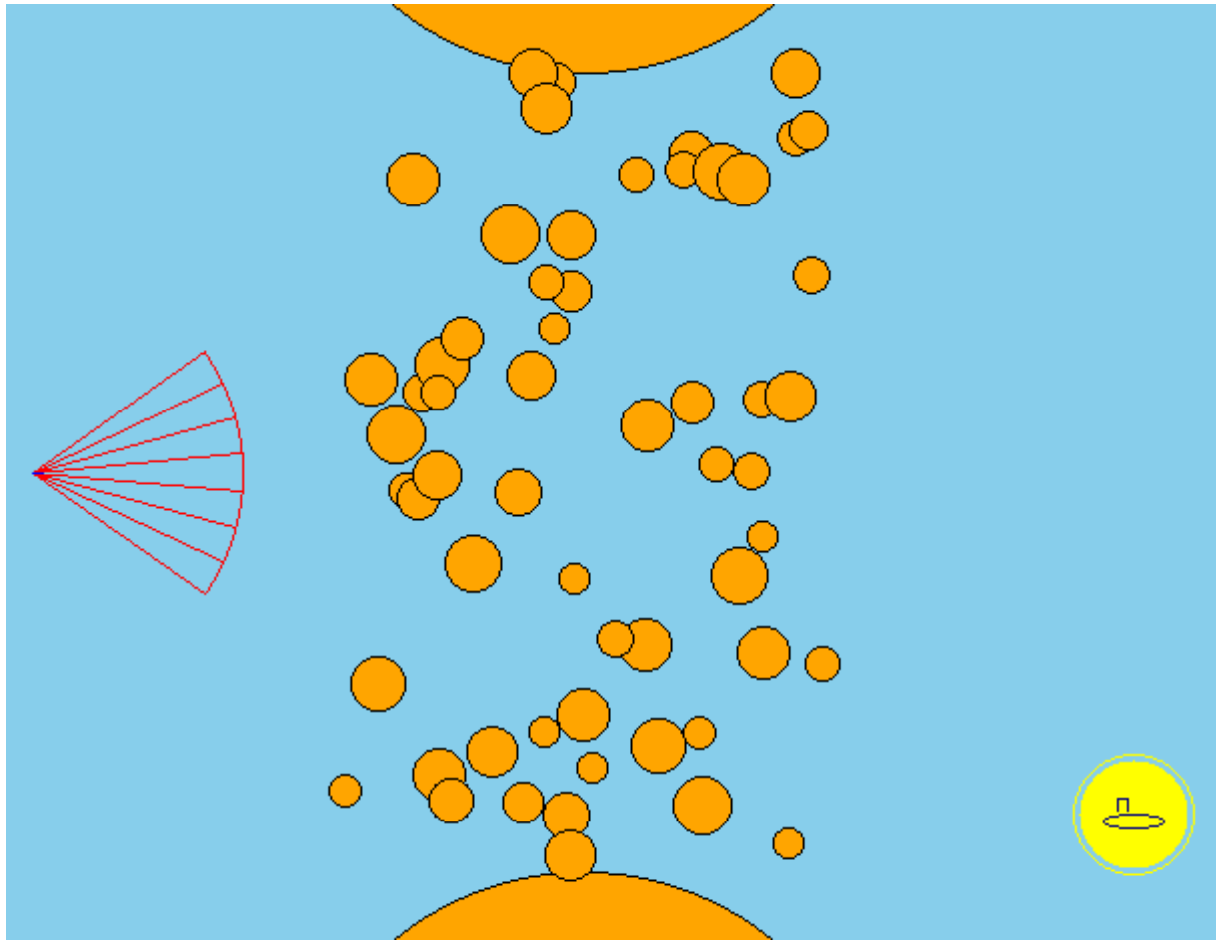
Blue: learn turns only

Red: learn turn and speed

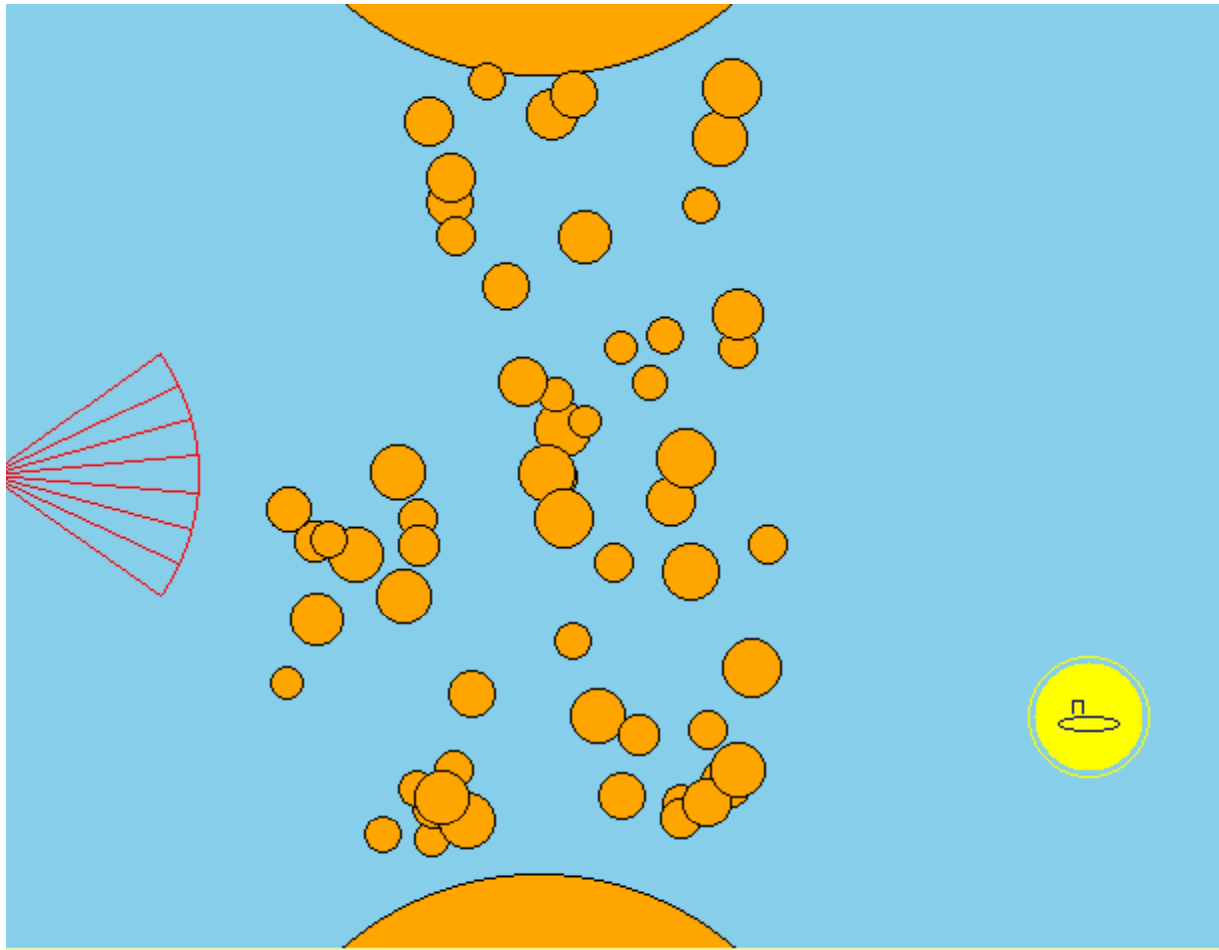
Humans make more effective use of training examples. But Q-learner gets to near 100% success.



Full Q learner/1500 episodes

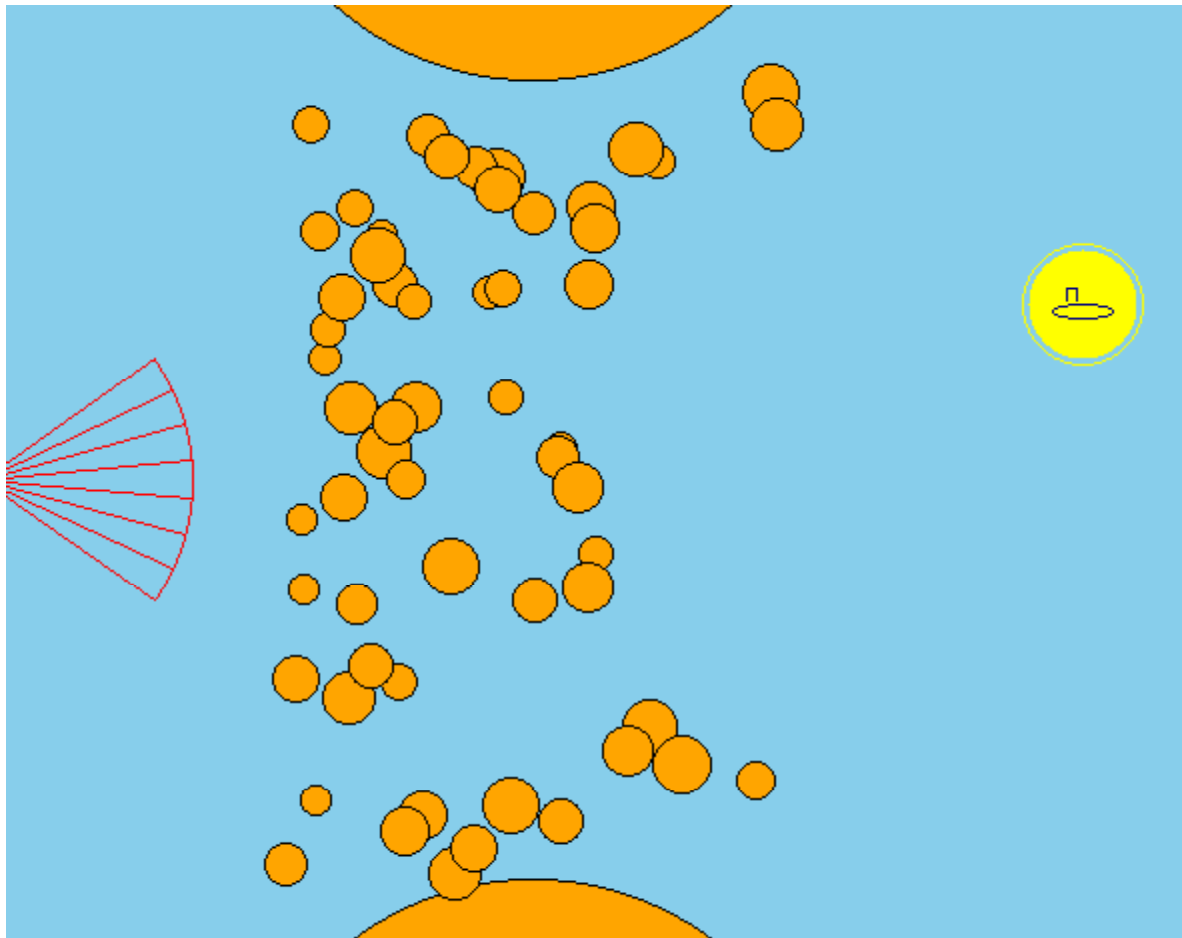


Full Q learner/10000 episodes

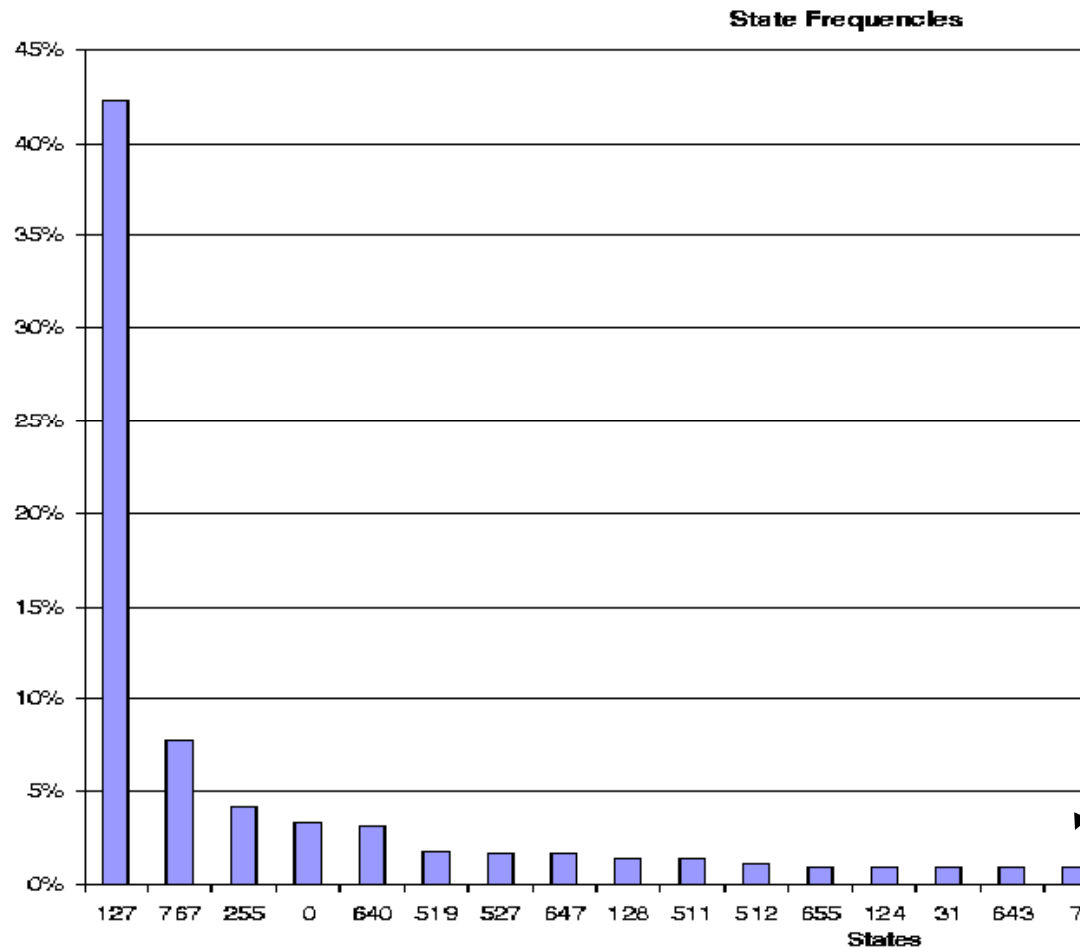




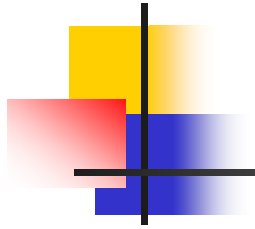
Full Q learner/failure after 10K



Why learning takes so long



States
where
3 or fewer
of the 163
action choices
are correct!





Lessons from machine learning

- Task is hard because states in which action choice is critical occur less than 5% of the time.
- Long sequence of moves makes credit assignment hard; using cheap approximation to global value function is useful.
- Staged learning makes task significantly easier.
- Need a locally non-deceptive reward function to speed up training.



Task Questions

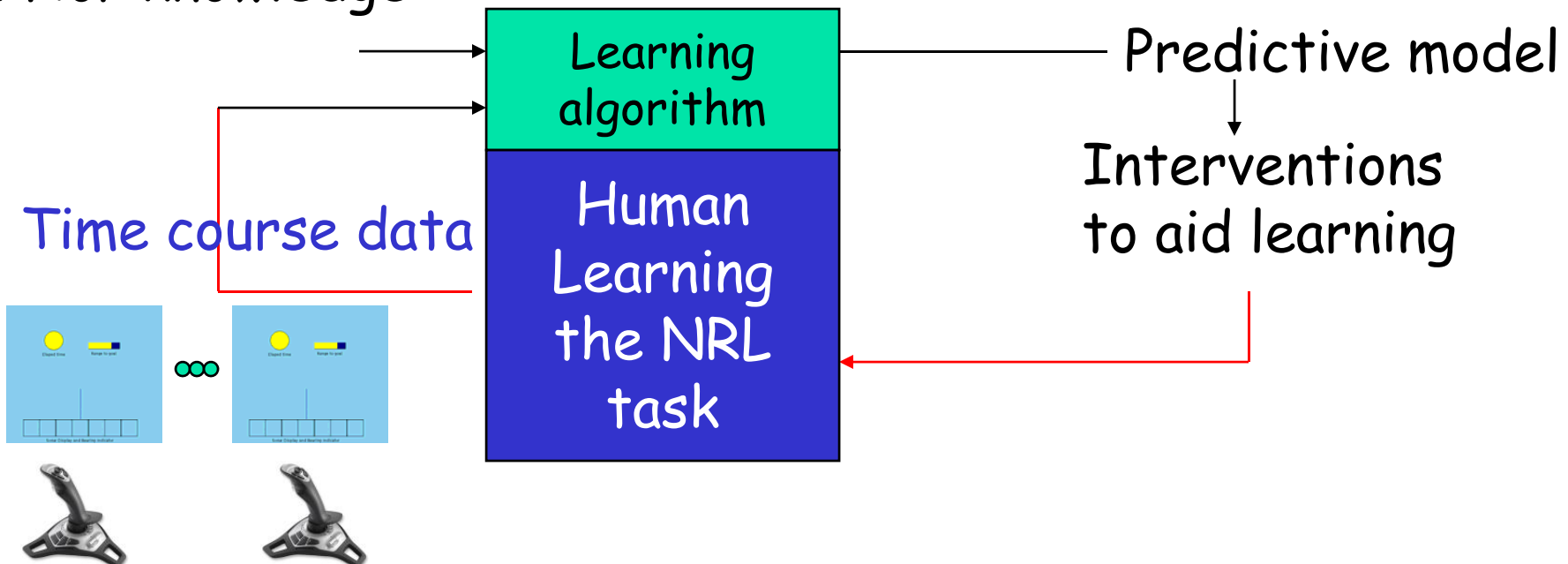
- 
- 
- Is the game hard? Is it hard for machines? What is the source of complexity?
 - Why does human performance plateau out at 80%? Is that a feature of the human learning system or the game? Can machine learners achieve higher levels of competence?
 - Can we understand why humans learn/fail to learn the task? Can we detect inability to learn early enough to intervene?
 - How can we actively shape human learning on this task?

Tracking human learning

Extract strategy and study its evolution over time

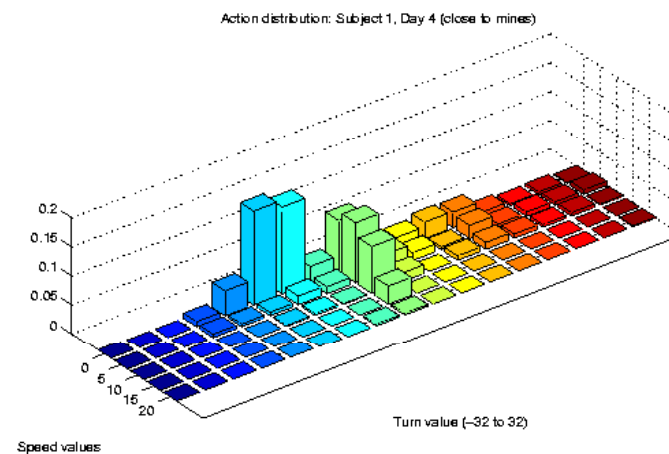
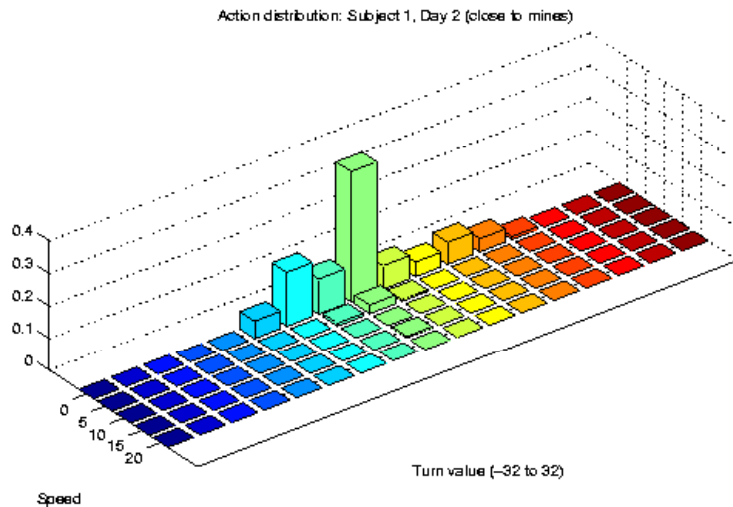
Prior knowledge

Time course data



Challenges

- High-dimensionality of visual data (11 dimensions spanning a space of size 10^{14})
- Large volumes of data
- Noise in data
- Non-stationarity: policies change over time





Embedded learner design

- Modeling
 - Use raw visual-motor data stream to induce policies/strategies.
- Learning
 - Direct models: lookup table mapping sensors at time t and action at $t-1$ to distribution of actions at time t . (1st order Markov model)
- Decision-making
 - Compute “derivative” of the policies over time, and use it (1) to classify learner and select interventions, (2) to build behaviorally equivalent models of subjects



Surely, this can't work!

- There are 10^{14} sensor configurations possible in the NRL Navigation task.
- However, there are between 10^3 to 10^4 of those configurations actually observed by humans in a training run of 600 episodes.
- Exploit sparsity in sensor configuration space to build a direct model of the subject.



Segmentation

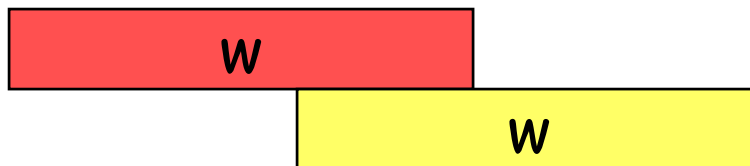
- Strategy computed on a window of size w ,

$$\Pi(i, i + w)$$

- *A lookup table mapping sensors to distributions on actions*

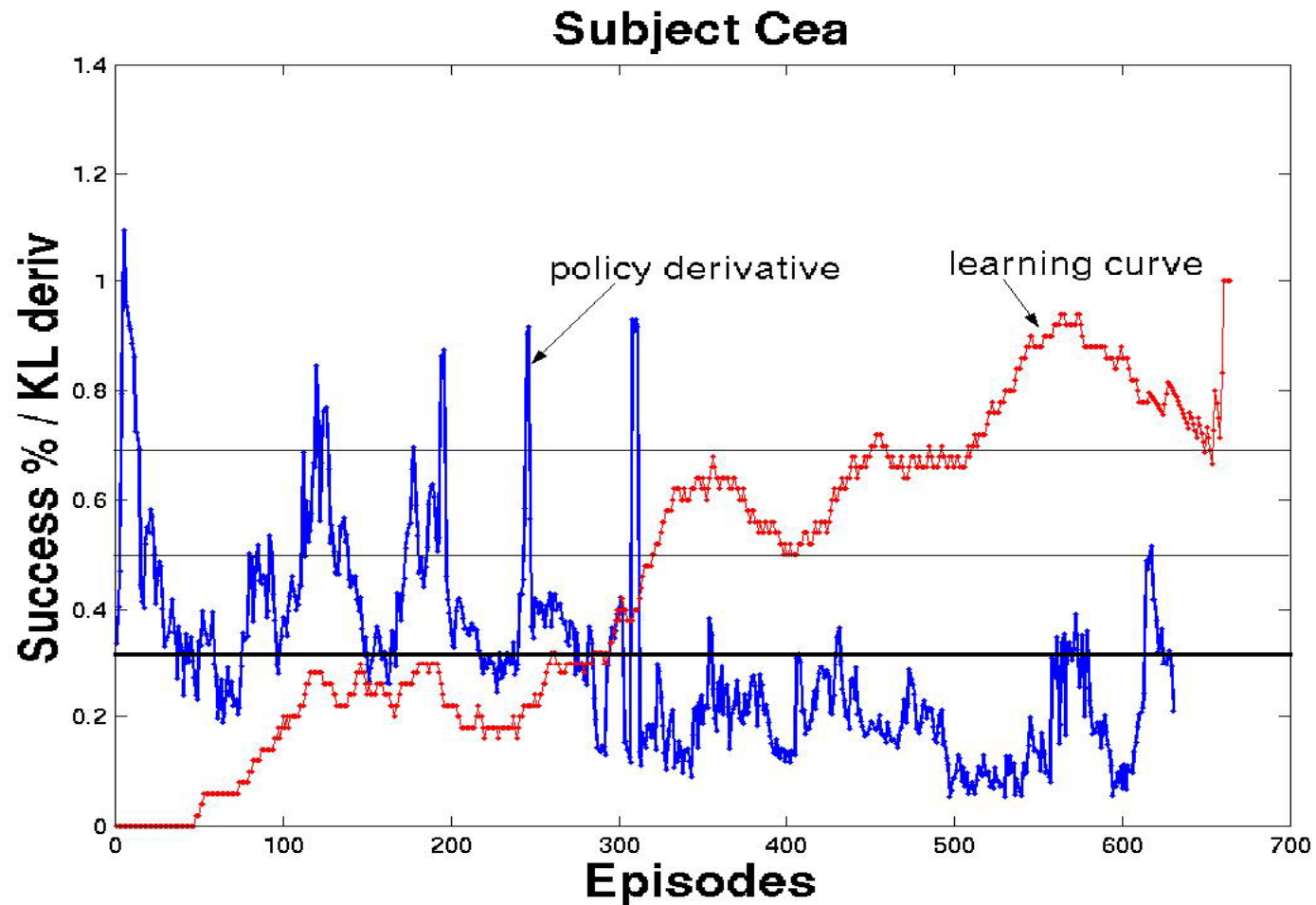
- Distance function between strategies: KL-derivative

$$\Delta_P(\Pi(i, i + w), \Pi(i + w - s, i + 2w - s))$$



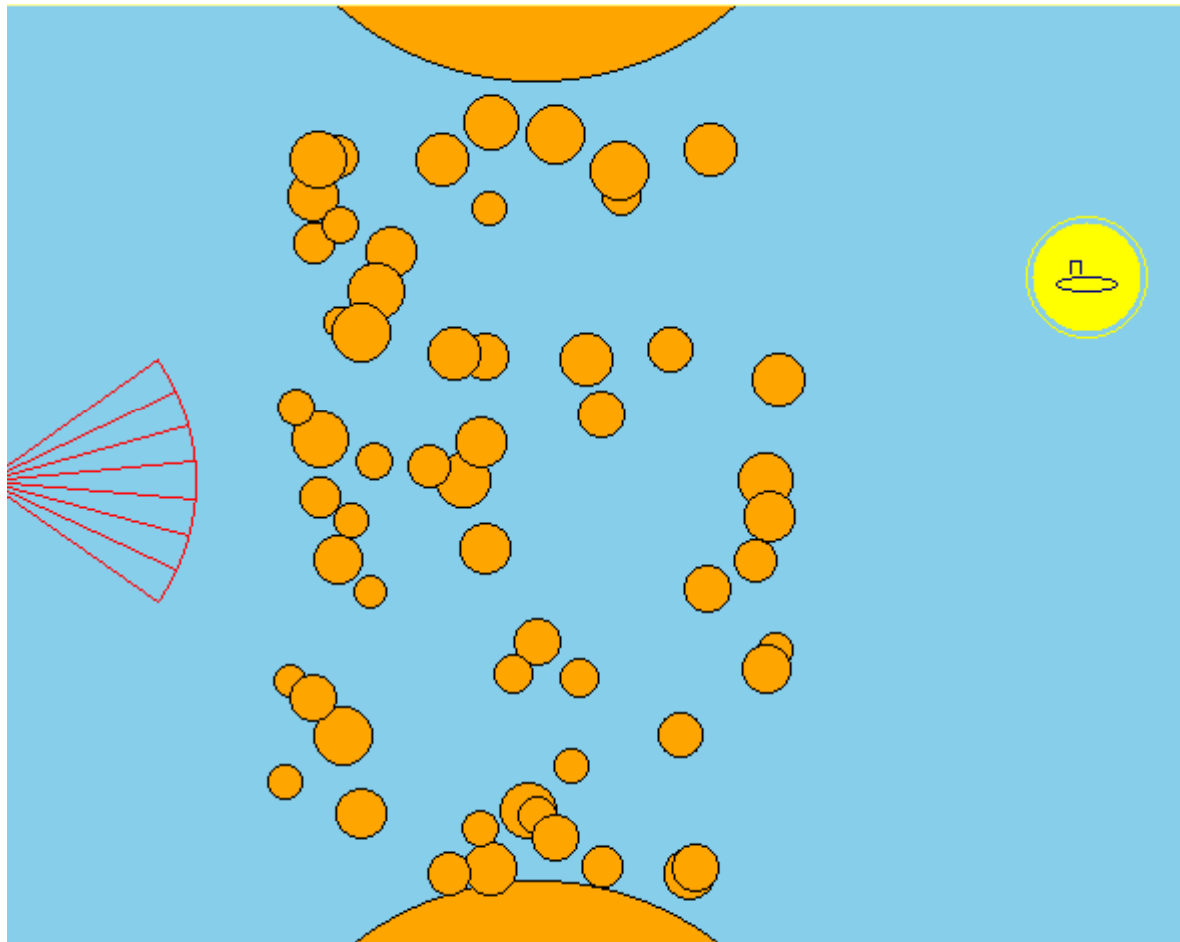
Overlap = s

Results: model derivative



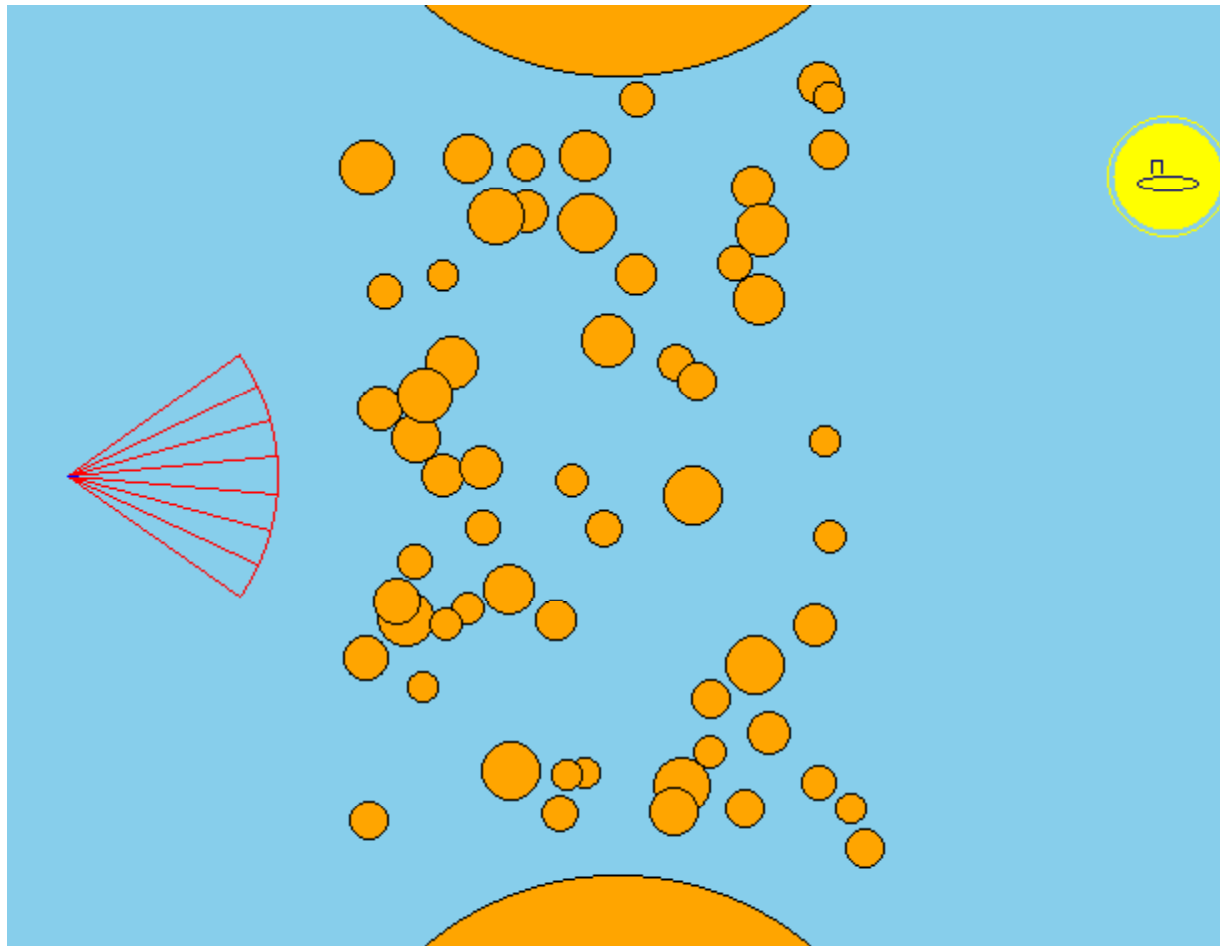


Before shift (episode 300)

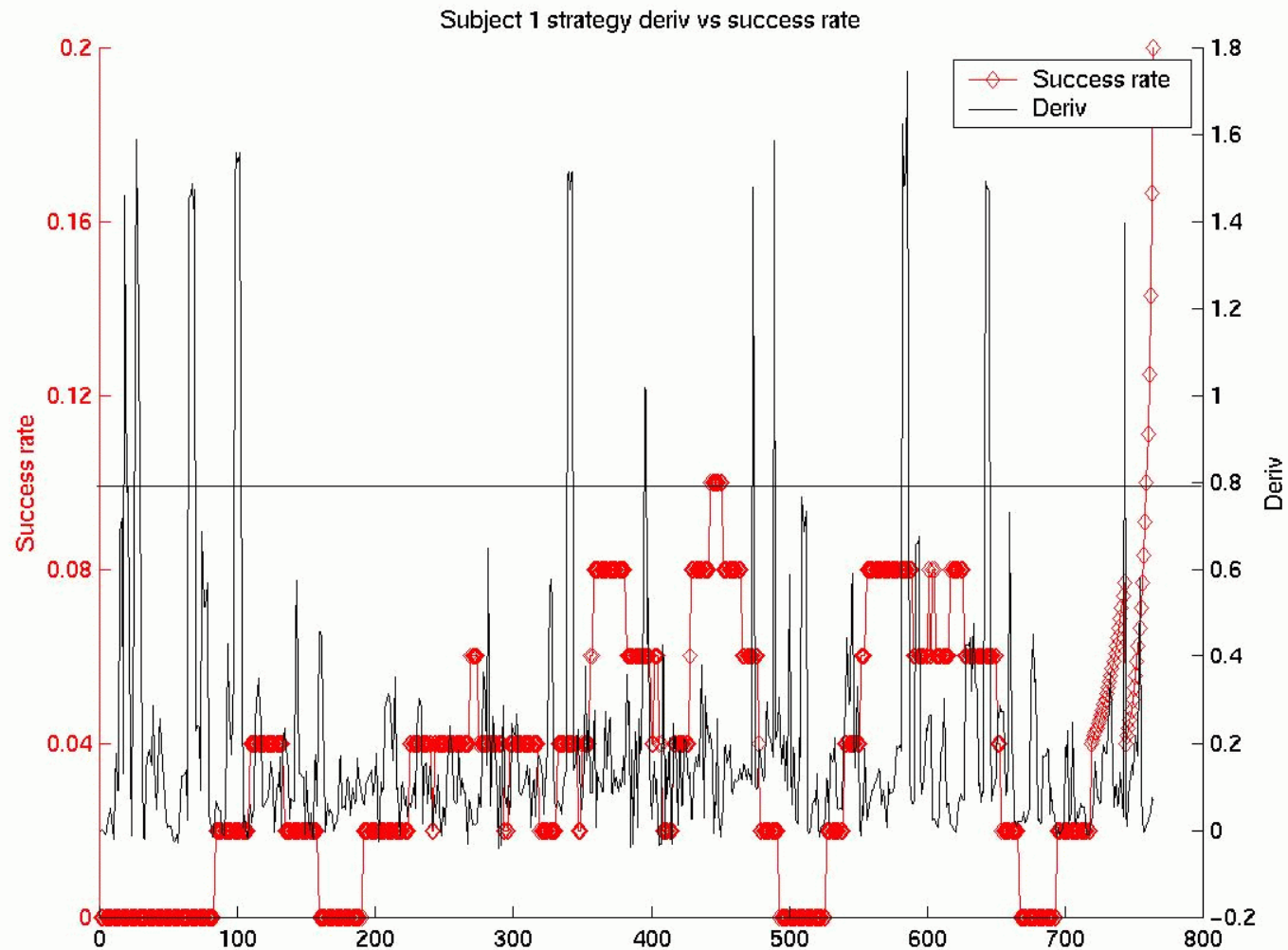




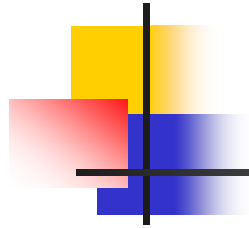
After shift (episode 320)



Model derivative for Hei



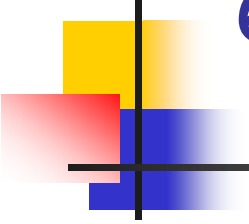
Siruguri and Subramanian, 2002



How humans learn

- Subjects have relatively static periods of action policy choice punctuated by radical shifts.
- Successful learners have conceptual shifts during the first part of training; unsuccessful ones keep trying till the end of the protocol!

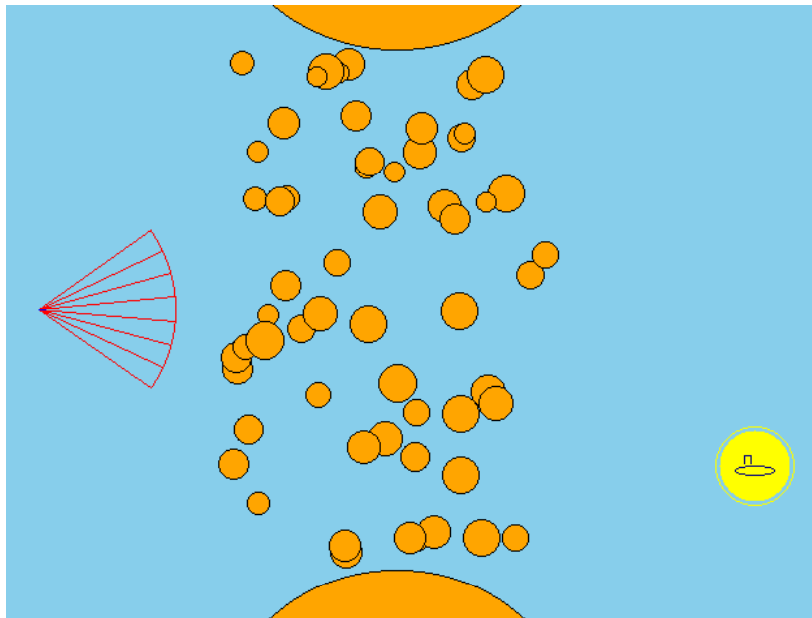
Generating behaviorally equivalent models



- To compute action a associated with current sensor configuration s in a given segment,
 - take 100 neighbors of s in lookup table.
 - perform locally weighted regression (LWR) on these 100 (s,a) pairs.



Subject Cea: Day 5: 1



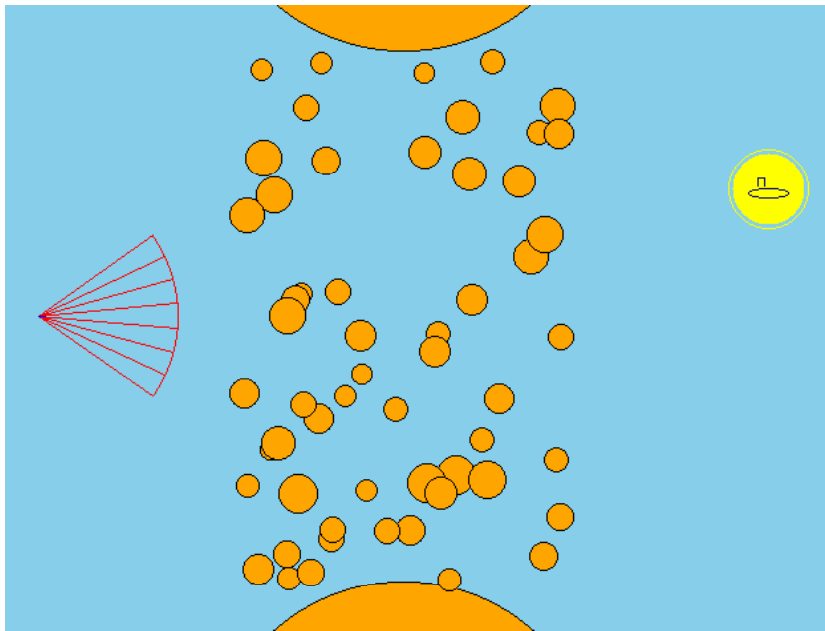
Subject



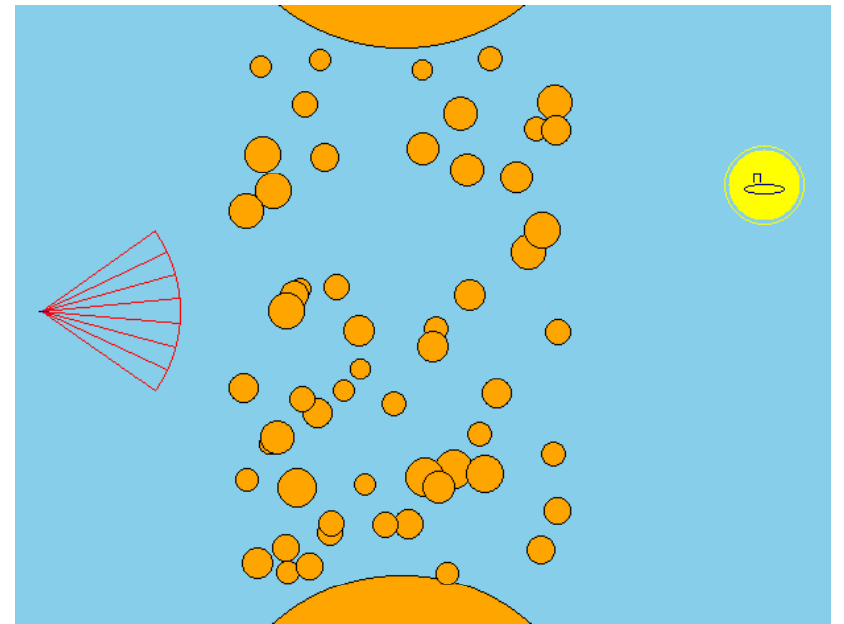
Model



Subject Cea: Day 5: 2



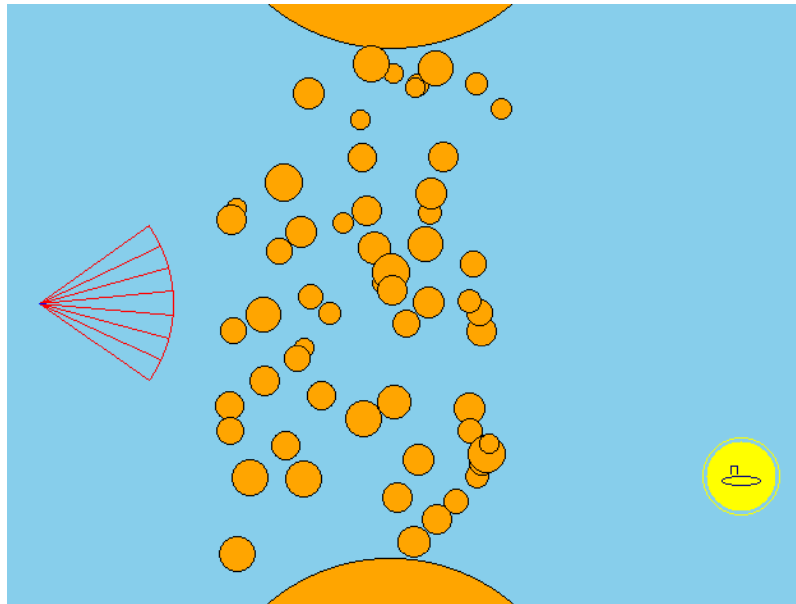
Subject



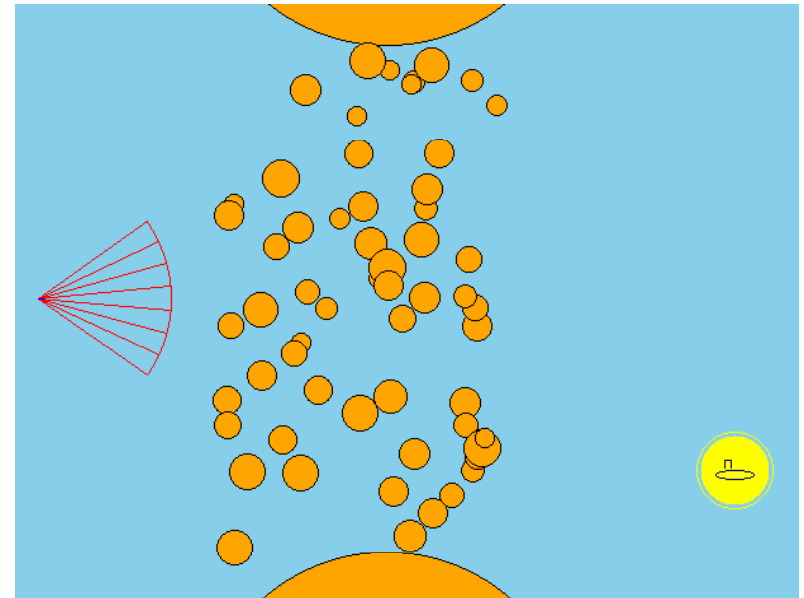
Model



Subject Cea: day 5: 3



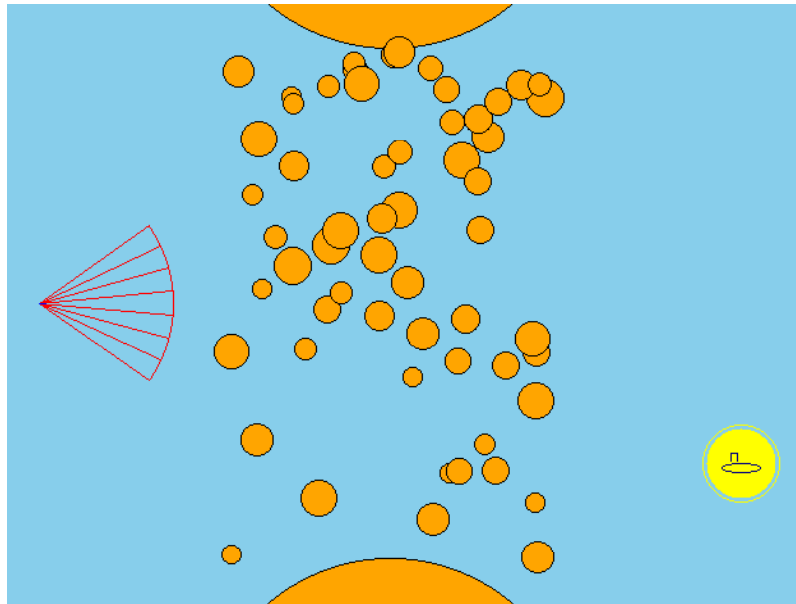
Subject



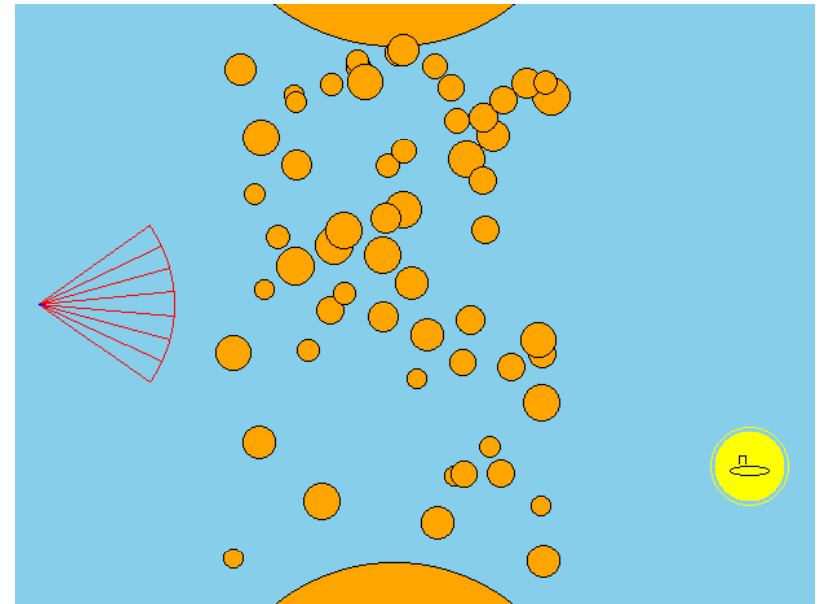
Model



Subject Cea: Day 5: 4

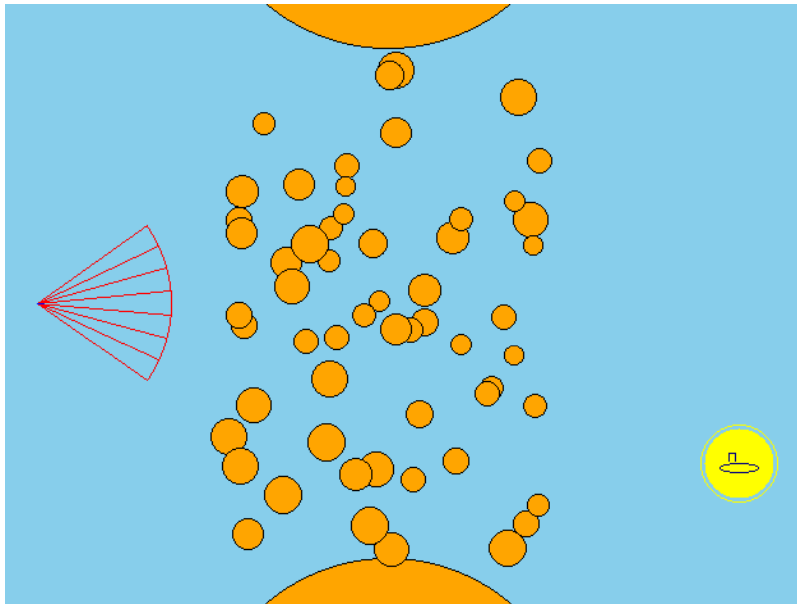


Subject

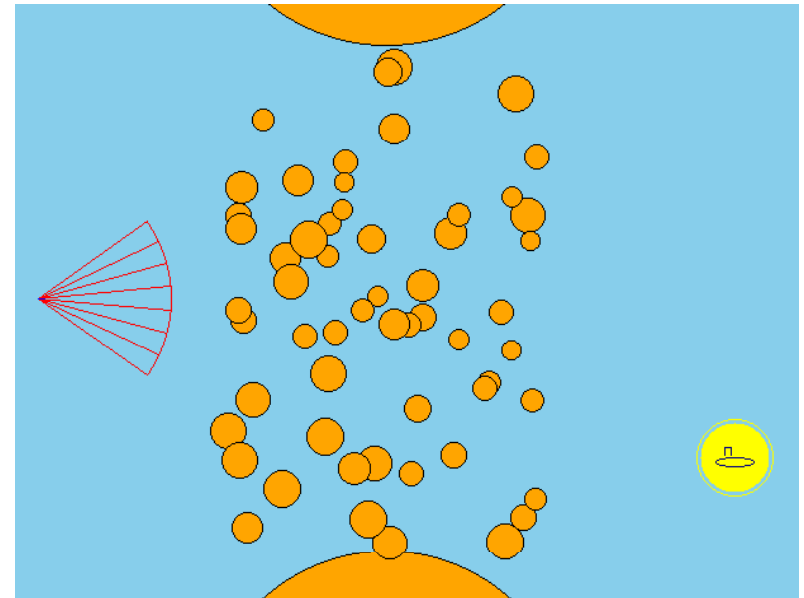


Model

Subject Cea: Day 5: 5

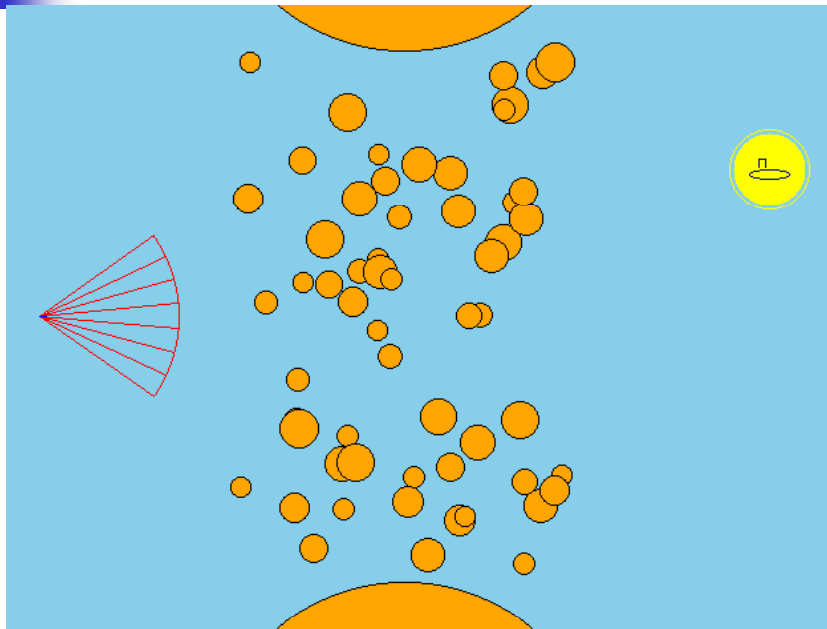


Subject

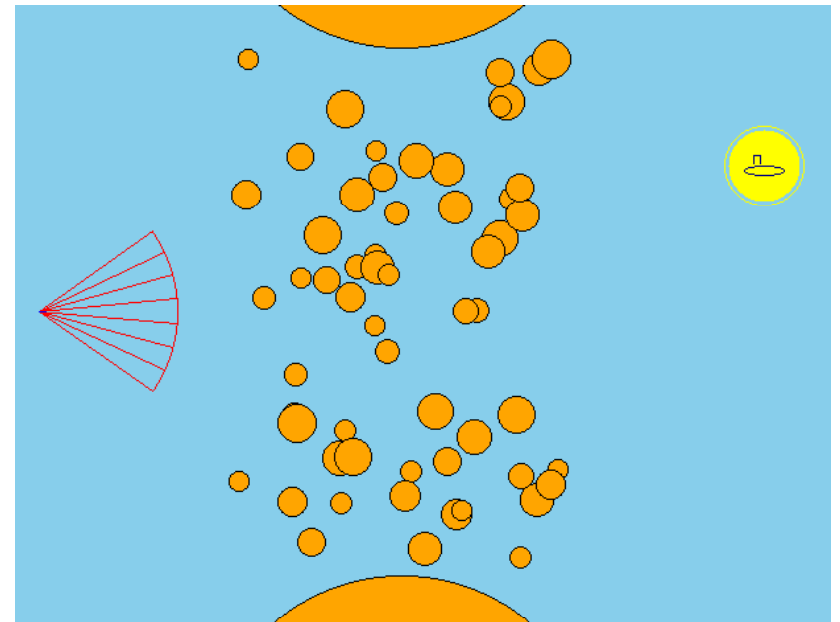


Model

Subject Cea: Day 5: 6



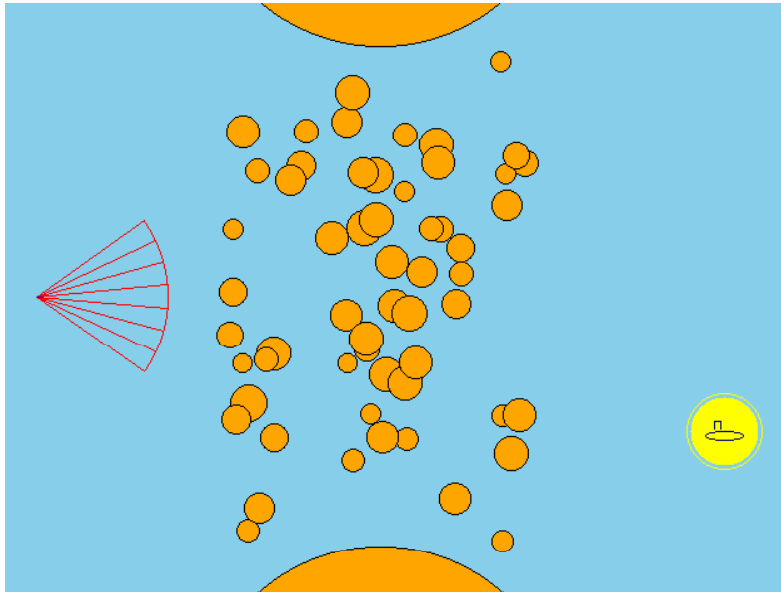
Subject



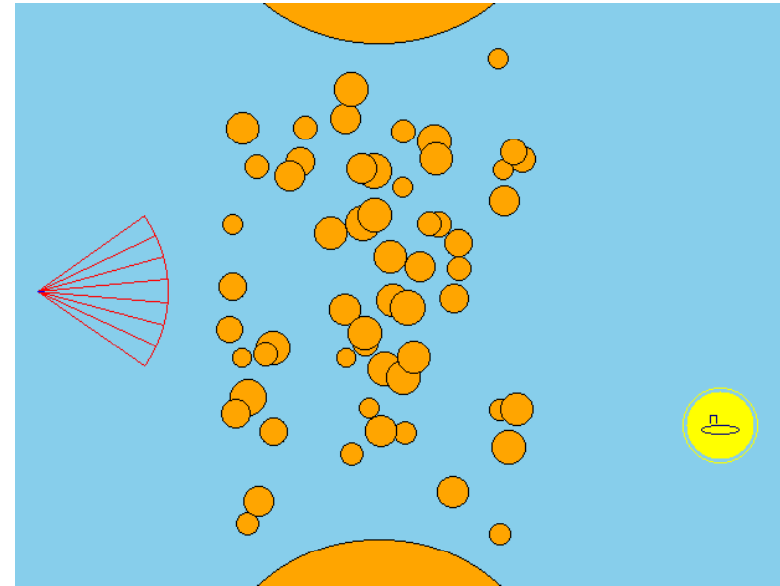
Model



Subject Cea: Day 5: 7



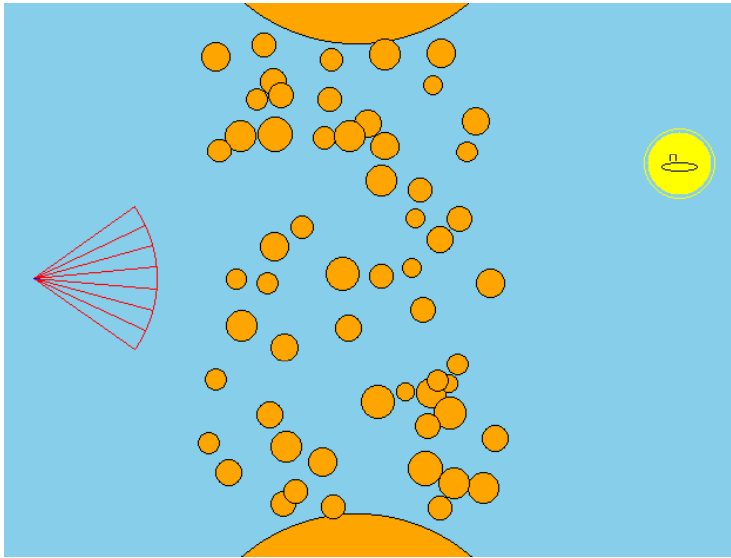
Subject



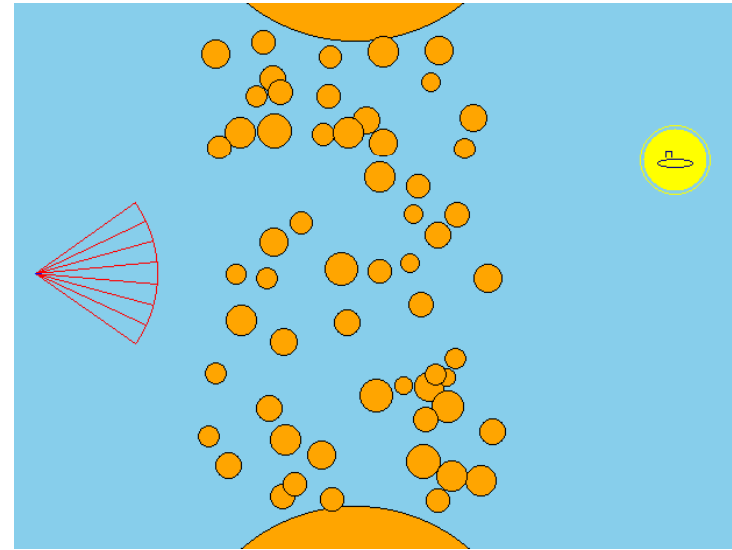
Model



Subject Cea: Day 5: 8



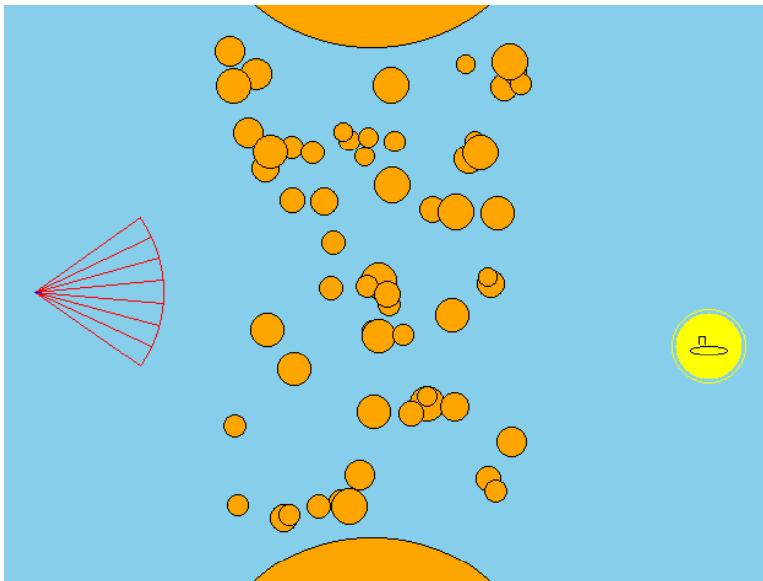
Subject



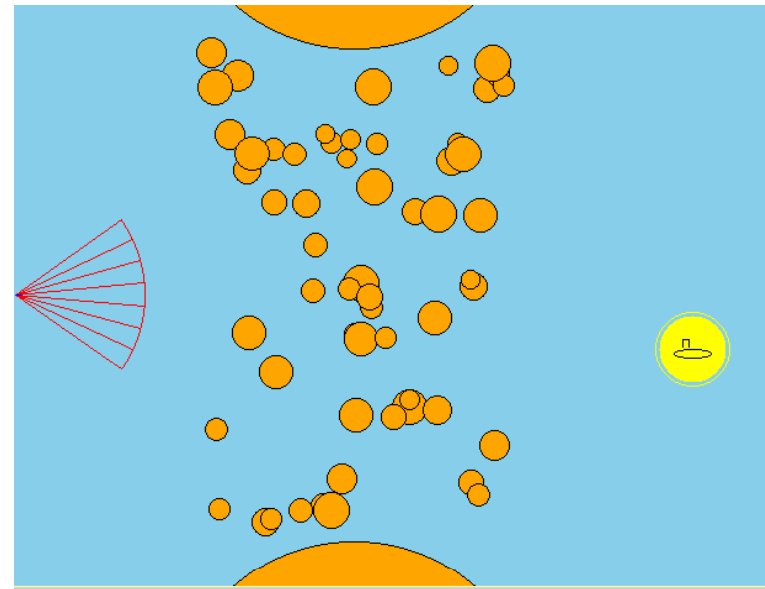
Model



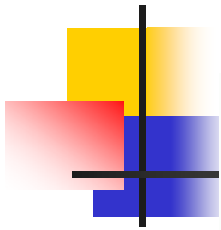
Subject Cea: Day 5: 9



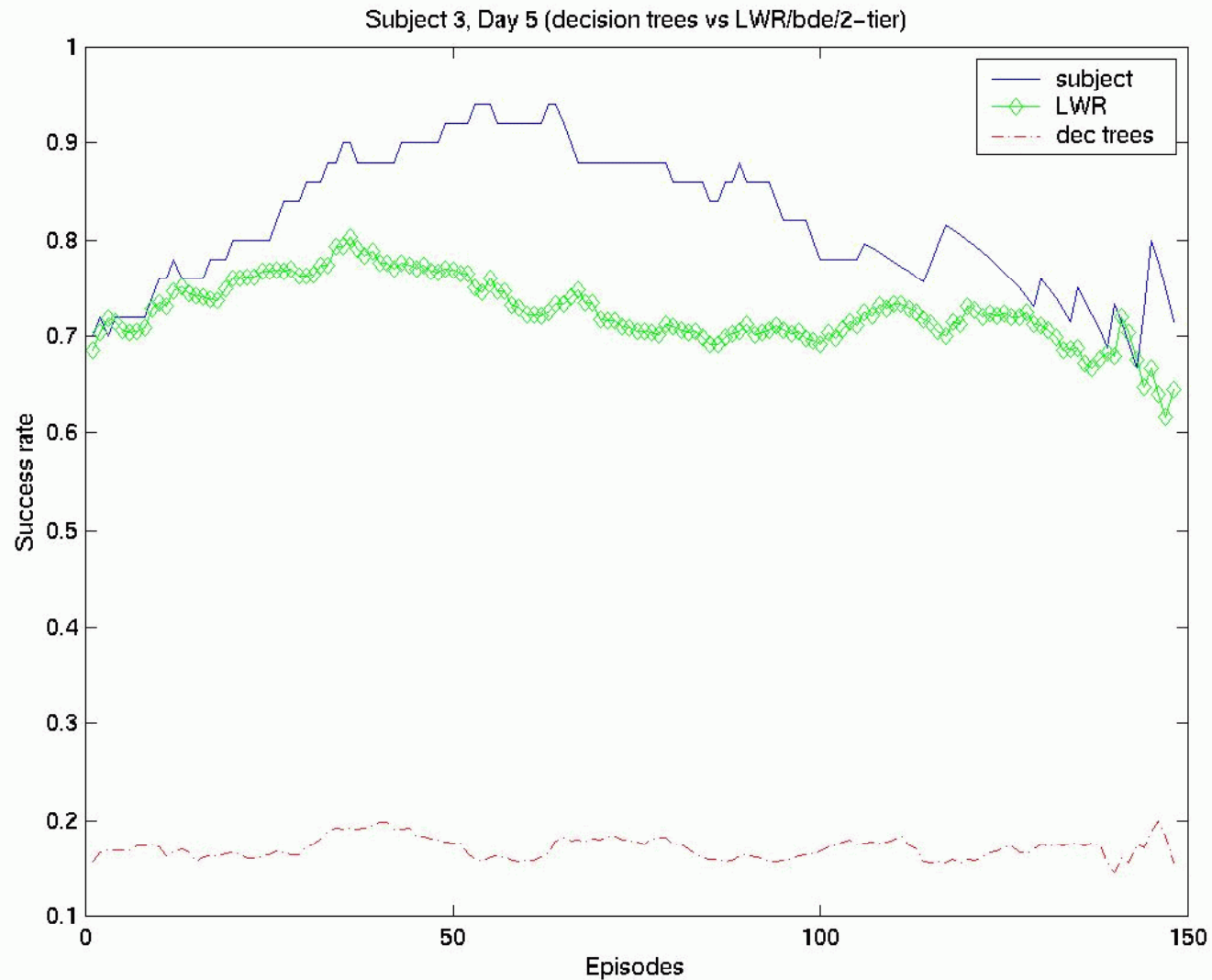
Subject



Model



Comparison with global methods





Summary (1)

- We can model subjects on the NRL task in **real-time**, achieving excellent fits to their learning curves, using the available visual-motor data stream.
- One of the first in cognitive science to directly use objective visual-motor performance data to derive high-level strategy models on a complex task.



Summary (2)

- Fast new algorithms for detecting change-points and building predictive stochastic models for massive, noisy, non-stationary, vector time series data.
- New approximations to speed up convergence of reinforcement learning for tasks with very long training sequences.

Current Direction

- Are there neural correlates to strategy shifts observed in the visual-motor data?

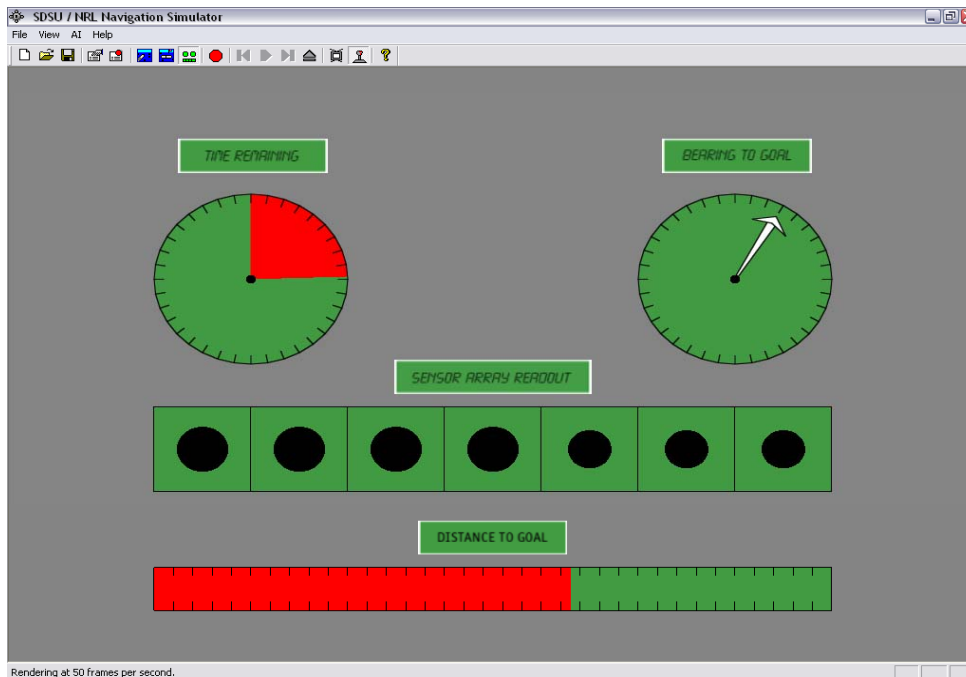
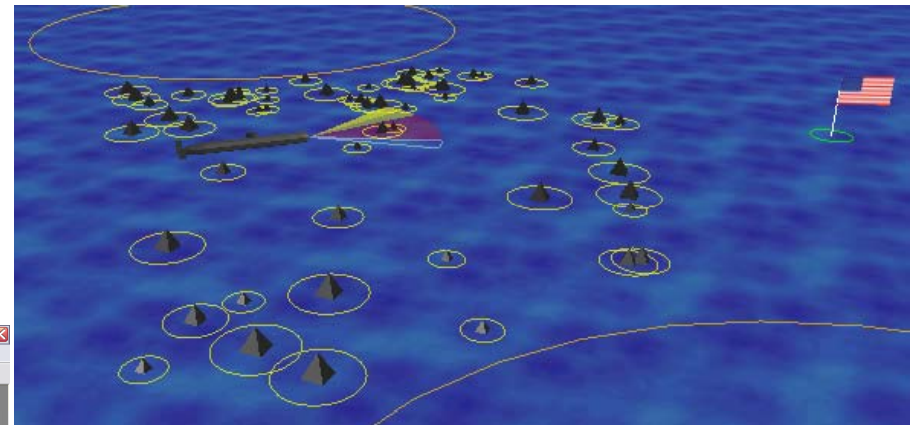




Task Questions

- Can we adapt training protocols in the NRL task by identifying whether subjects are struggling with strategy formulation or visual-motor control or both?
- Can we use analysis of EEG data gathered during learning as well as visual-motor performance data to correlate 'brain events' with 'visual-motor performance events'? Can this correlation separate subjects with different learning difficulties?

The (new) NRL Navigation Task

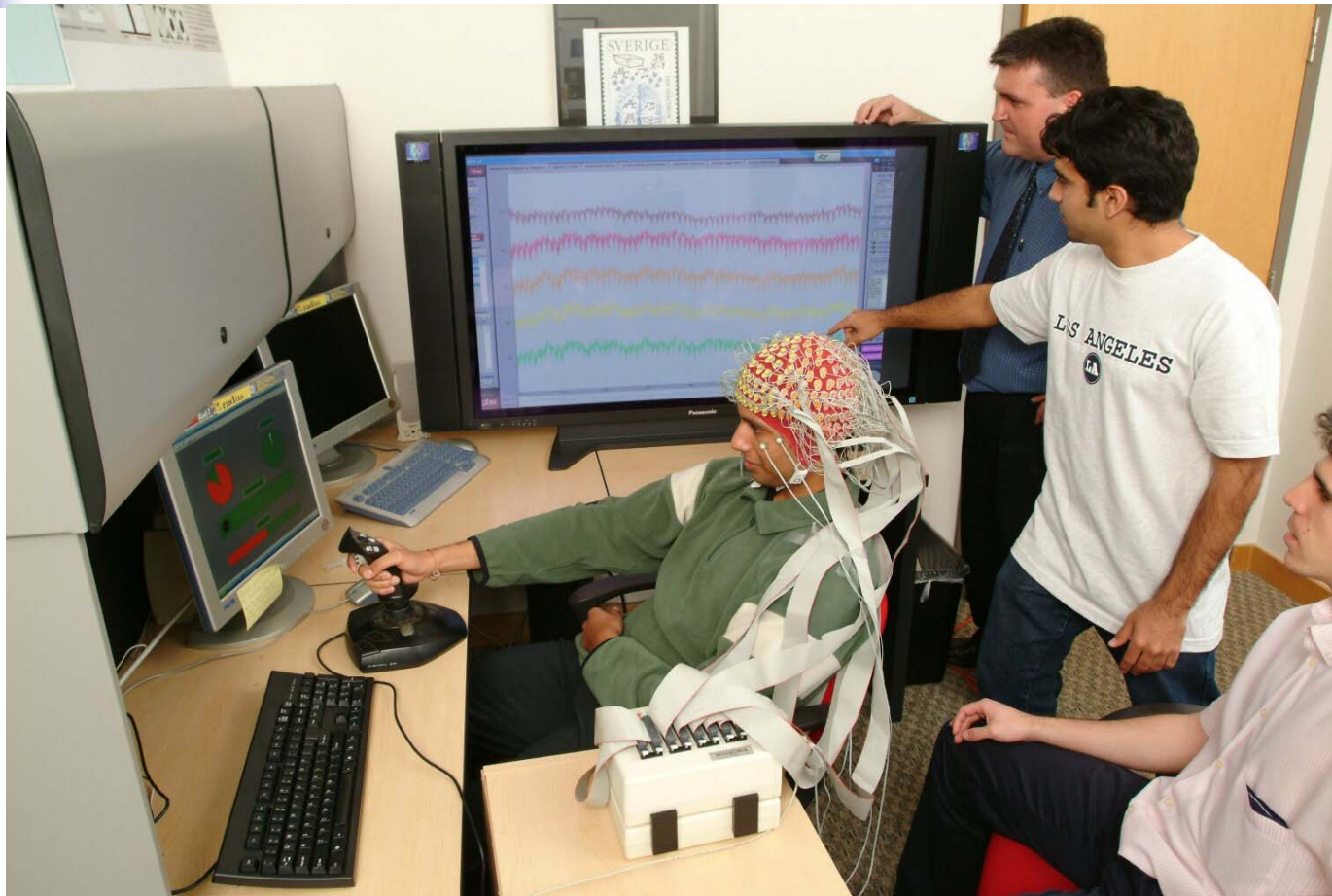


SDSU's and **NRL's**
Navigation Simulator v1.0

Created by Michael Kennedy and John Stricker
Cognitive Ergonomics Research Facility
San Diego State University

Copyright 1999. This program is protected under U.S. and international law.

Gathering performance data



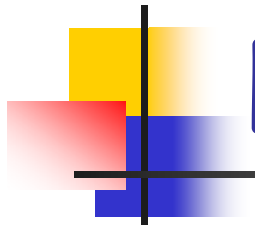
256 channel EEG recording



The coherence function

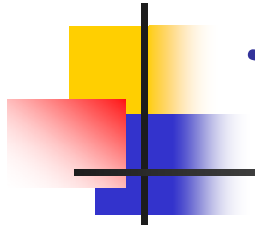
- *Coherence* provides the means to measure synchronous activity between two brain areas
- A function that calculates the normalized cross-power spectrum, a measure of similarity of signal in the frequency domain

$$C_{xy}(f) = \frac{|S_{xy}(f)|^2}{[S_{xx}(f)S_{yy}(f)]}$$

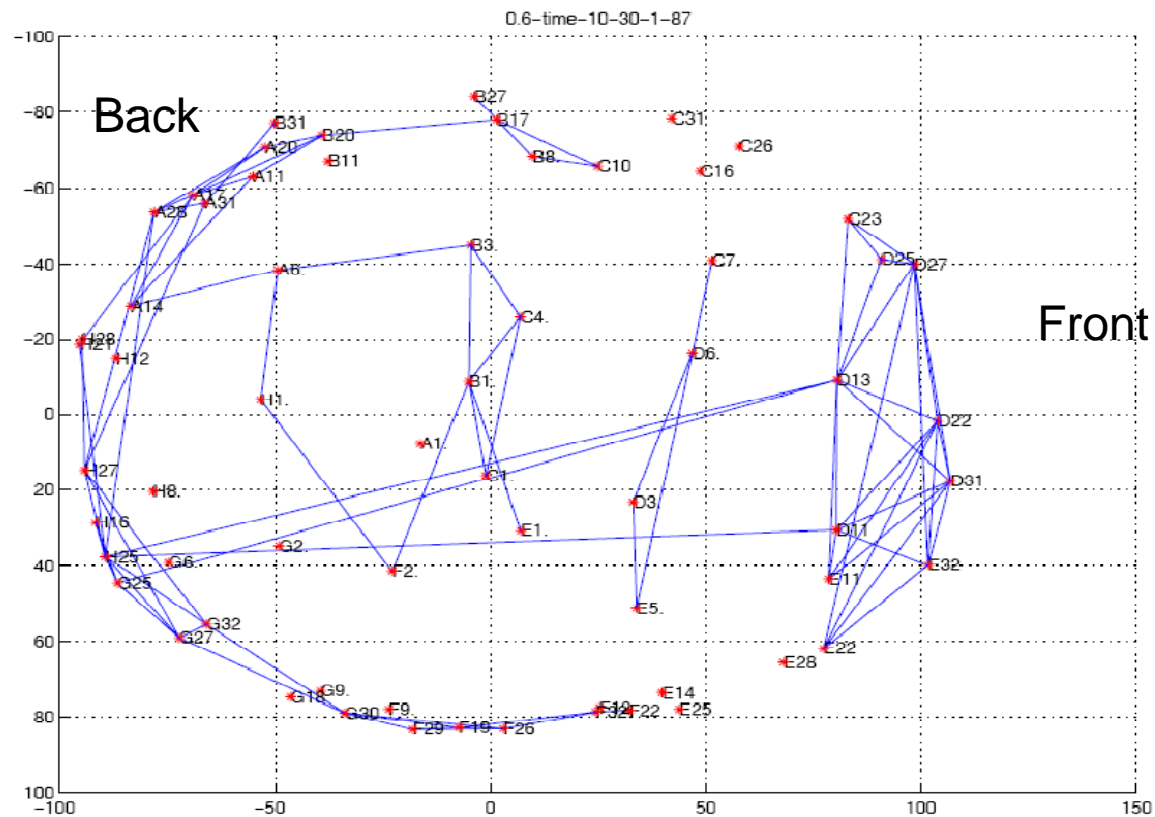


Frequency bands

- Get a map of connections in each band (>0.6)
 - Δ (0-5 Hz)
 - θ (5-9 Hz)
 - α (9-14 Hz)
 - β (14-30 Hz)
 - γ (40-52 Hz)
 - All Bands

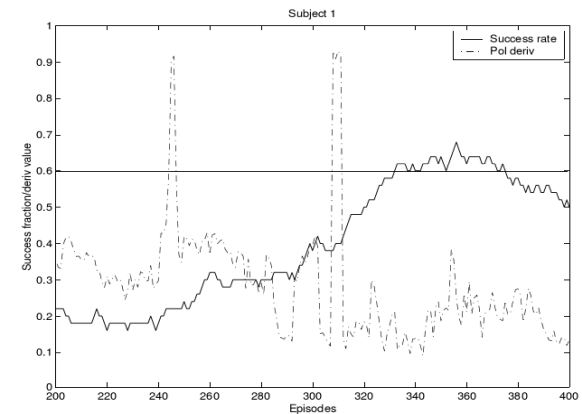
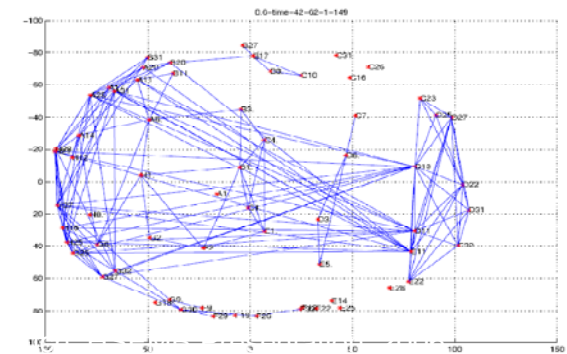


Topological coherence map

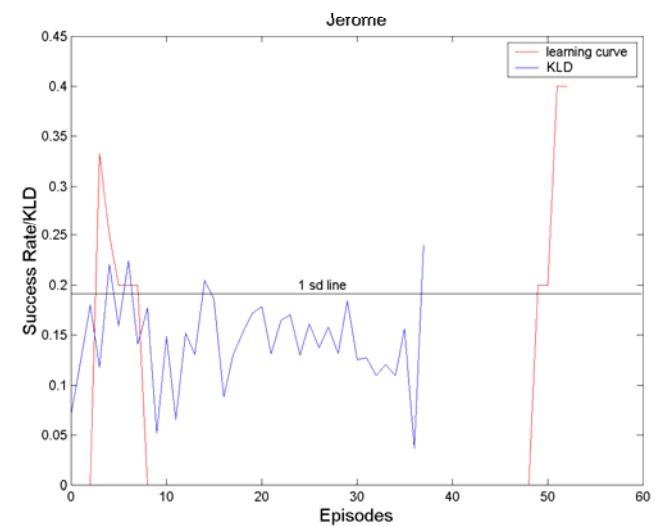
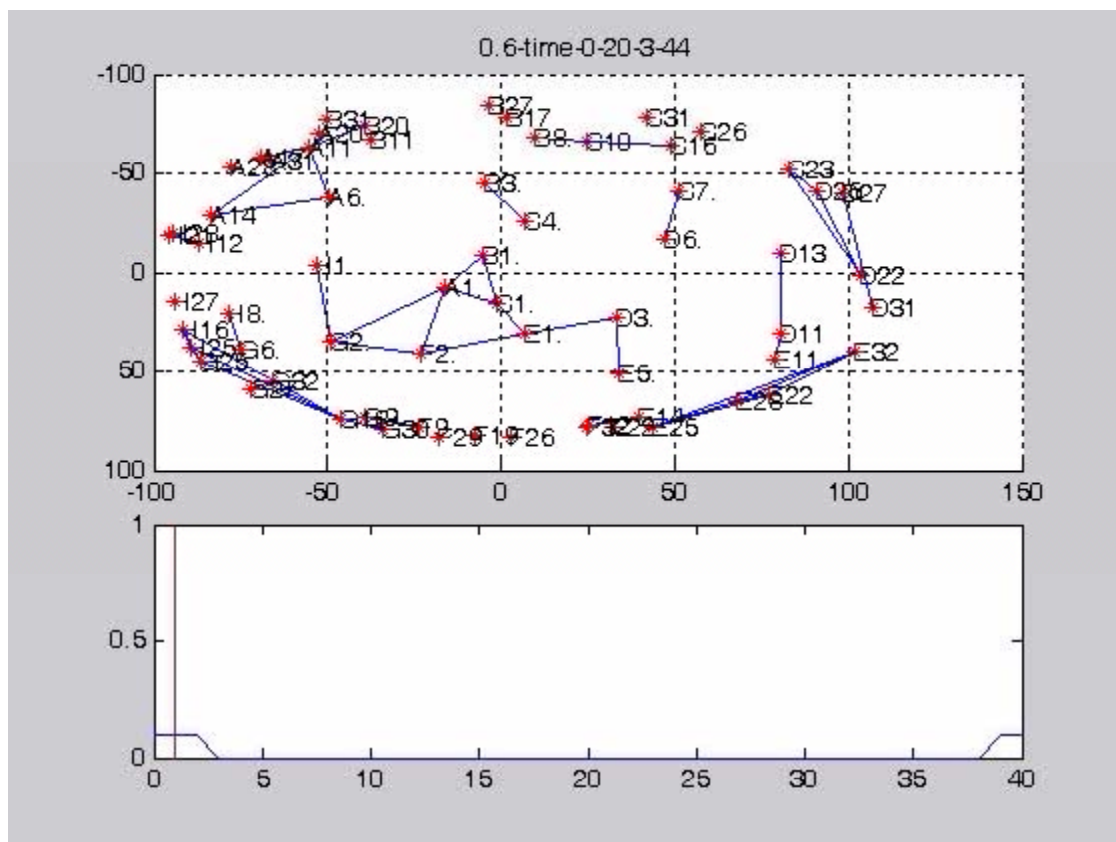


Performance data

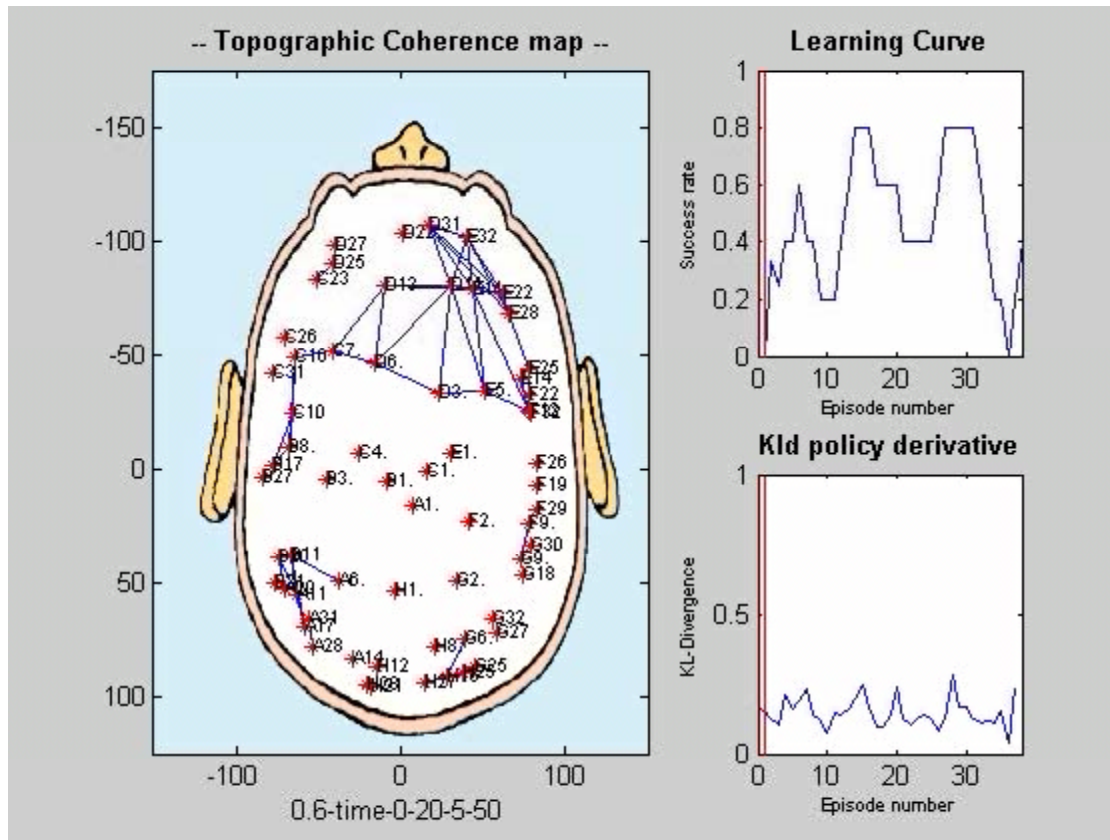
- What we record:
 - EEG (256 Channel)
 - Motor Data from Joystick
 - Visual Data from Simulator
- What we compute:
 - Coherence Maps
 - Strategy derivatives
 - Learning Curves



Subject J



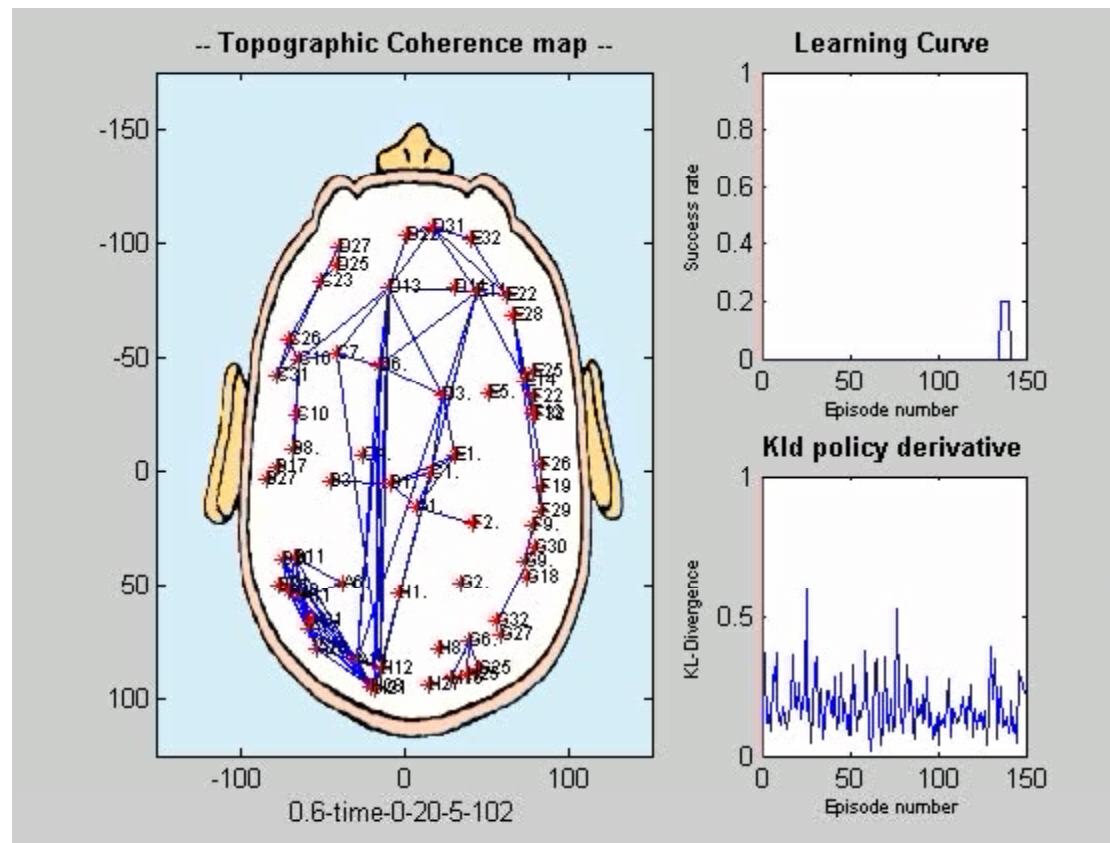
Subject G



Subject is a near-expert performer

Subject is in skill refinement phase

Subject V



Subject
never
learned a
good strategy

It wasn't
for lack
of trying..



Results

- There are distinct EEG coherence map signatures associated with different learning difficulties
 - Lack of strategy
 - Shifting between too many strategies
- The coherence maps of successful learners show shifts between dense to sparse interregional cortical synchrony. It is indicative of shifts between strategy formulation to strategy retrieval and visual-motor loop tuning.
- We are conducting experiments on more subjects to confirm these findings. (14 subjects so far, and more are being collected right now.)

Diagnosis of learning deficits



"My problem has always been an overabundance of alpha waves."



Publications

- Human Learning and the Neural Correlates of Strategy Formulation, F. Baluch, D. Subramanian and G. Zouridakis, 23rd Annual Conference on Biomedical Engineering Research, 2006.
- Understanding Human Learning on Complex Tasks by Functional Brain Imaging, D. Subramanian, R. Bandyopadhyay and G. Zouridakis, 20th Annual Conference on Biomedical Engineering Research, 2003.
- Tracking the evolution of learning on a visuo-motor task Devika Subramanian and Sameer Siruguri, Technical report TR02-401, Department of Computer Science, Rice University, August 2002.
- Tracking the evolution of learning on a visuo-motor task Sameer Siruguri, Master's thesis under the supervision of Devika Subramanian, May 2001.
- State Space Discretization and Reinforcement Learning, S. Griffin and D. Subramanian, Technical report, Department of Computer Science, Rice University, June 2000.
- Inducing hybrid models of learning from visuo-motor data, *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, Philadelphia, PA, 2000.
- Modeling individual differences on the NRL Navigation task, *Proceedings of the 20th Annual Conference of the Cognitive Science Society*, Madison, WI, 1998 (with D. Gordon).
- A cognitive model of learning to navigate, *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, Stanford, CA, 1997 (with D. Gordon).
- Cognitive modeling of action selection learning, *Proceedings of the 18th Annual Conference of the Cognitive Science Society*, San Diego, 1996 (with D. Gordon)

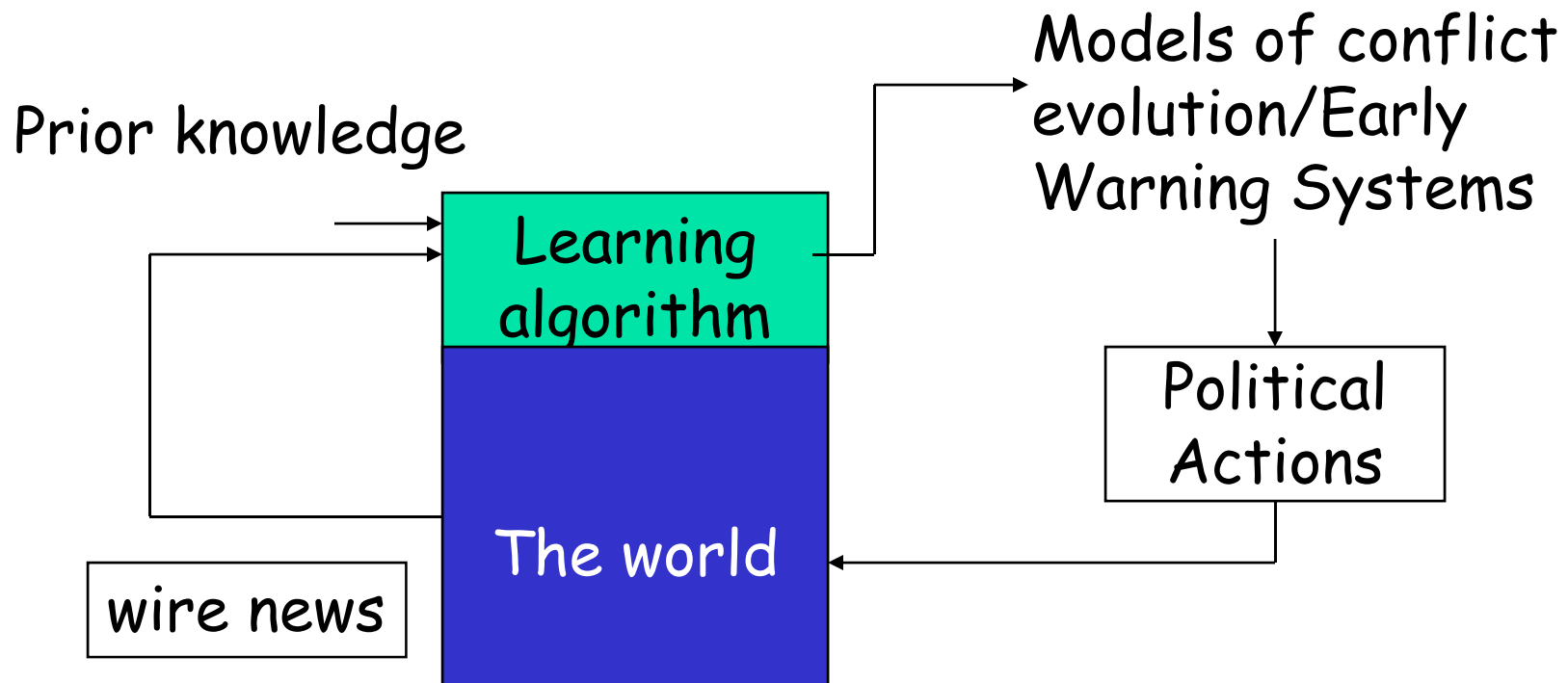


Roadmap of talk

- Four case-studies
 - Unknown system, changing dynamics
 - Tracking human learning on a complex visual-motor task.
 - Predicting the evolution of international conflict.
 - Unknown system, customization needed
 - Learning disruptions in metabolic processes in cancer cells.
 - Designing customized compiler optimization sequences for application programs.



Forecasting conflict

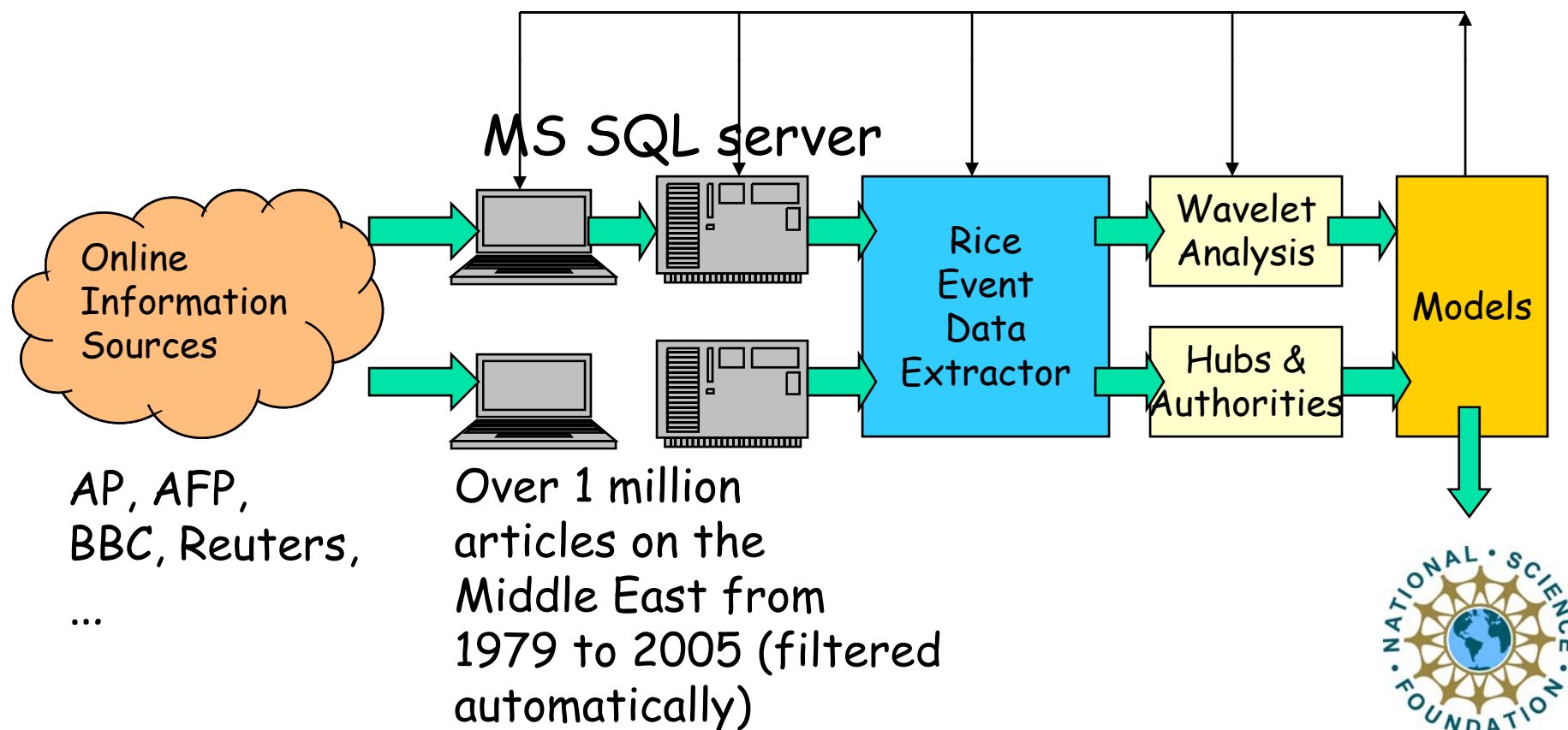




Task Question

- Is it possible to monitor news media from regions all over the world over extended periods of time, extracting low-level **events** from them, and piece them together to automatically track and predict conflict in all the regions of the world?

The Ares project





Embedded learner design

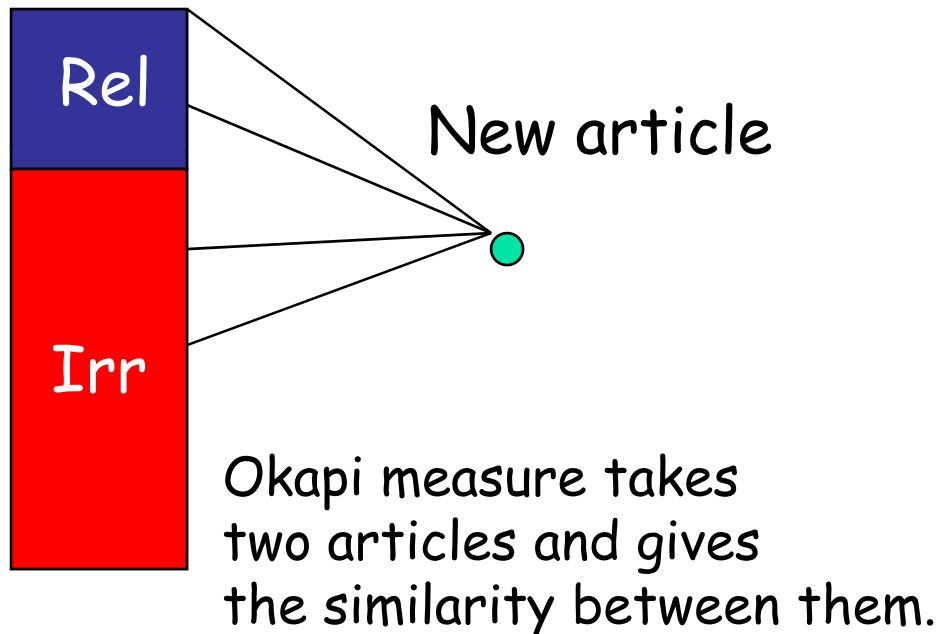
- Modeling
 - Identify relevant stories, extract event data from them, build time series models and graph-theoretic models.
- Learning
 - Identifying regime shifts in events data, tracking evolution of militarized interstate disputes (MIDs) by hubs/authorities analysis of events data
- Decision-making
 - Issuing early warnings of outbreak of MIDs



Identifying relevant stories

- Only about 20% of stories contain events that are to be extracted.
 - The rest are interpretations of conflictual events (e.g., op-eds), or are events not about conflict (e.g., sports, human-interest stories, etc.).
- We have trained Naïve Bayes (precision 86% and recall 81%) & Okapi classifiers (precision 93% and recall 87%) using a labeled set of 180,000 stories from Reuters.
- Surprisingly difficult problem!
 - Lack of large labeled data sets;
 - Poor transfer to other sources (AP/BBC)
 - The category of “event containing stories” is not well-separated from others, and changes with time

Okapi classifier



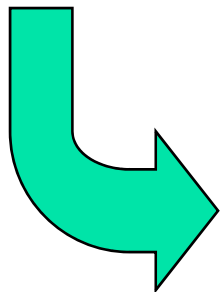
- Reuters data set:
relevant categories are GVIO, GDIP, G13;
irrelevant categories: 1POL, 2ECO, 3SPO, ECAT, G12, G131, GDEF, GPOL

Decision rule: sum of top N Okapi scores in Rel set > sum of top N Okapi scores in Irr set
then classify as rel; else irr



Event extraction

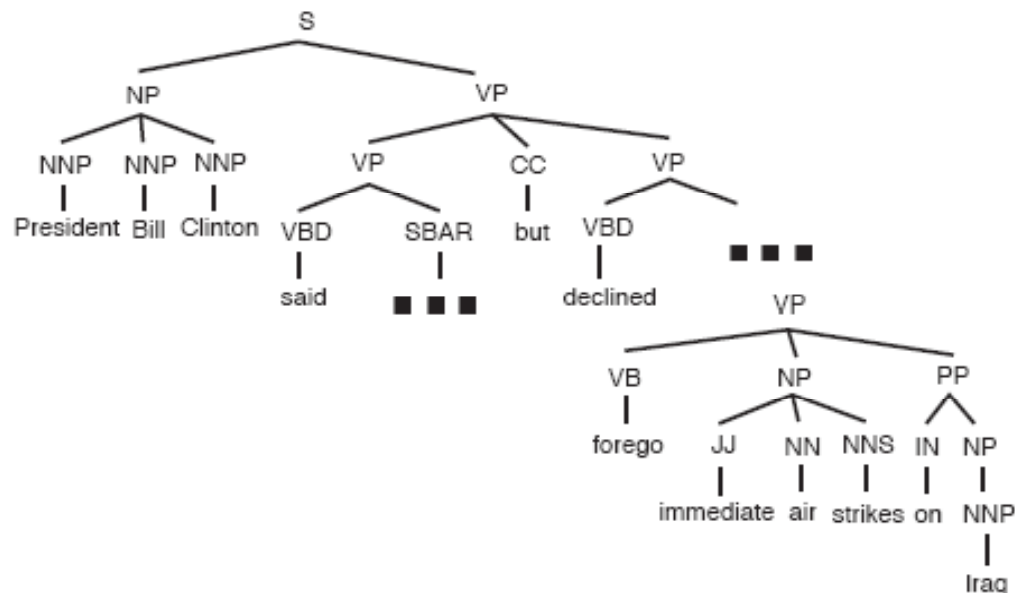
"President Bill Clinton said on Monday the United States sought no confrontation with Iraqi President Saddam Hussein but declined to say whether that meant he would forego immediate air strikes on Iraq."



Fragment	Event data
President Bill Clinton said on Monday the United States sought no confrontation with Iraqi President Saddam Hussein	USA Comment USA
the United States sought no confrontation with Iraqi President Saddam Hussein	USA Deny IRQ
President Bill Clinton declined to say whether that meant President Bill Clinton would forego immediate air strikes on Iraq	USA Comment USA
President Bill Clinton would forego immediate air strikes on Iraq	not part of event phrase (did not actually happen)

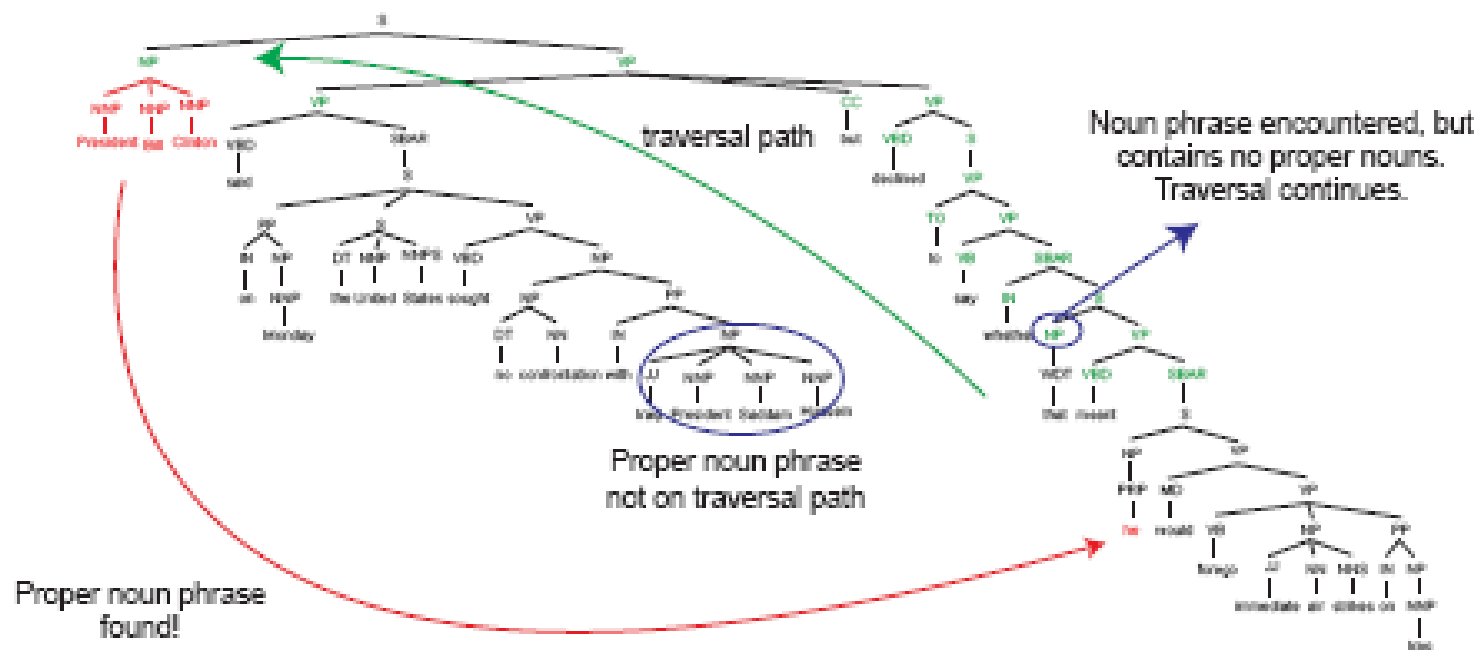
Parse sentence

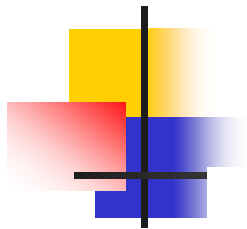
1. *President Bill Clinton said on Monday the United States sought no confrontation with Iraqi President Saddam Hussein but declined to say whether that meant he would forego immediate air strikes on Iraq.*



Klein and Manning parser

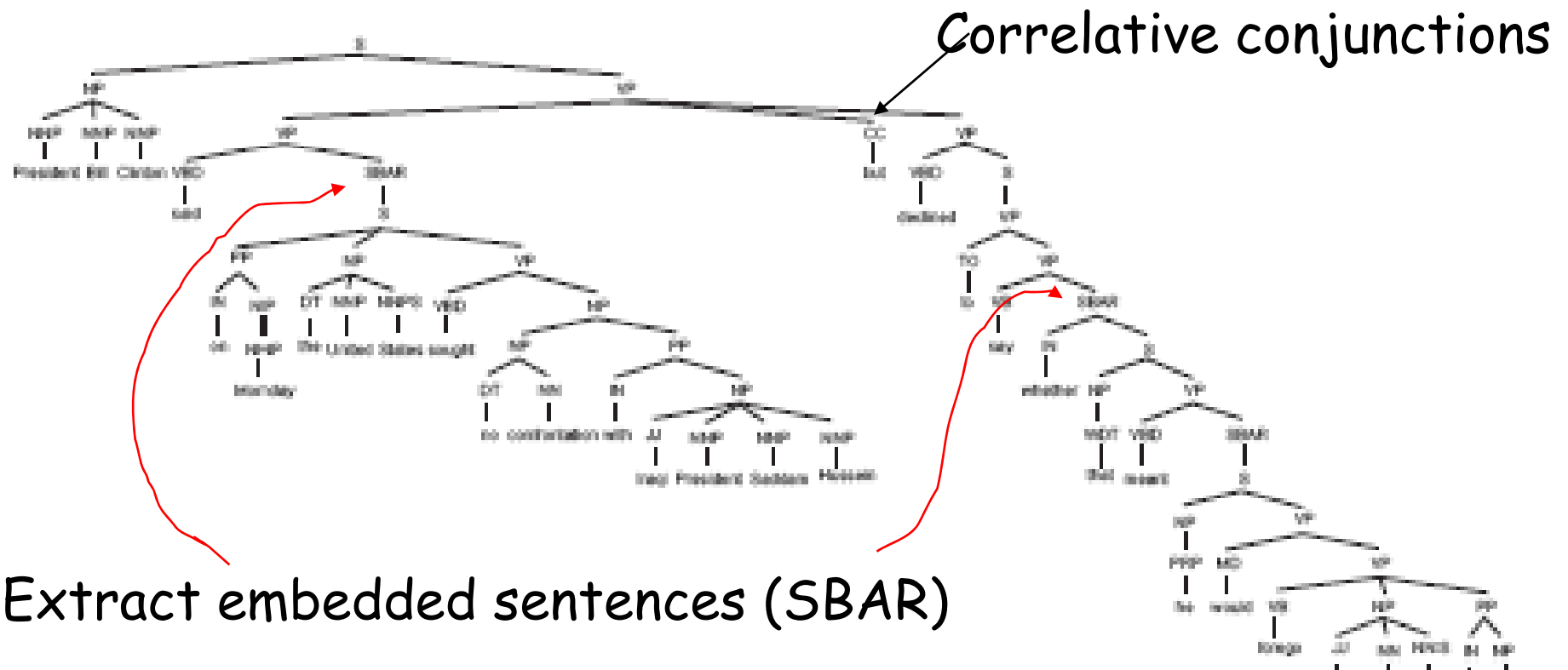
Pronoun de-referencing





Sentence fragmentation

"President Bill Clinton said on Monday the United States sought no confrontation with Iraqi President Saddam Hussein but declined to say whether that meant he would forego immediate air strikes on Iraq."





Conditional random fields

We extract who (actor) did what (event) to whom (target)

Actor & target labels

- countries, capitals, nationalities
- words in proper noun phrases
- actors occur before main verb
- actors at higher levels of tree
- targets occur after main verb
- targets at lower levels of tree

Event category labels

- specific event keywords
- words in main verb phrase
- specific parts of speech
- not modified by negative words
- part of event phrase

Not exactly the same as NER



Results

TABARI
is state
of the art
coder
in political
science

Table 1: Results for 22 (Force) category

Coder	Accuracy	Recall	Precision
TABARI	22%	7%	50%
TABARI with frag	20%	8%	83%
CRF	72%	70%	91%

Table 2: Results for 02 (Comment) category

Coder	Accuracy	Recall	Precision
TABARI	81%	31%	67%
TABARI with frag	88%	54%	93%
CRF	89%	96%	68%

200 Reuters sentences; hand-labeled with actor, target, and event codes (22 and 02).

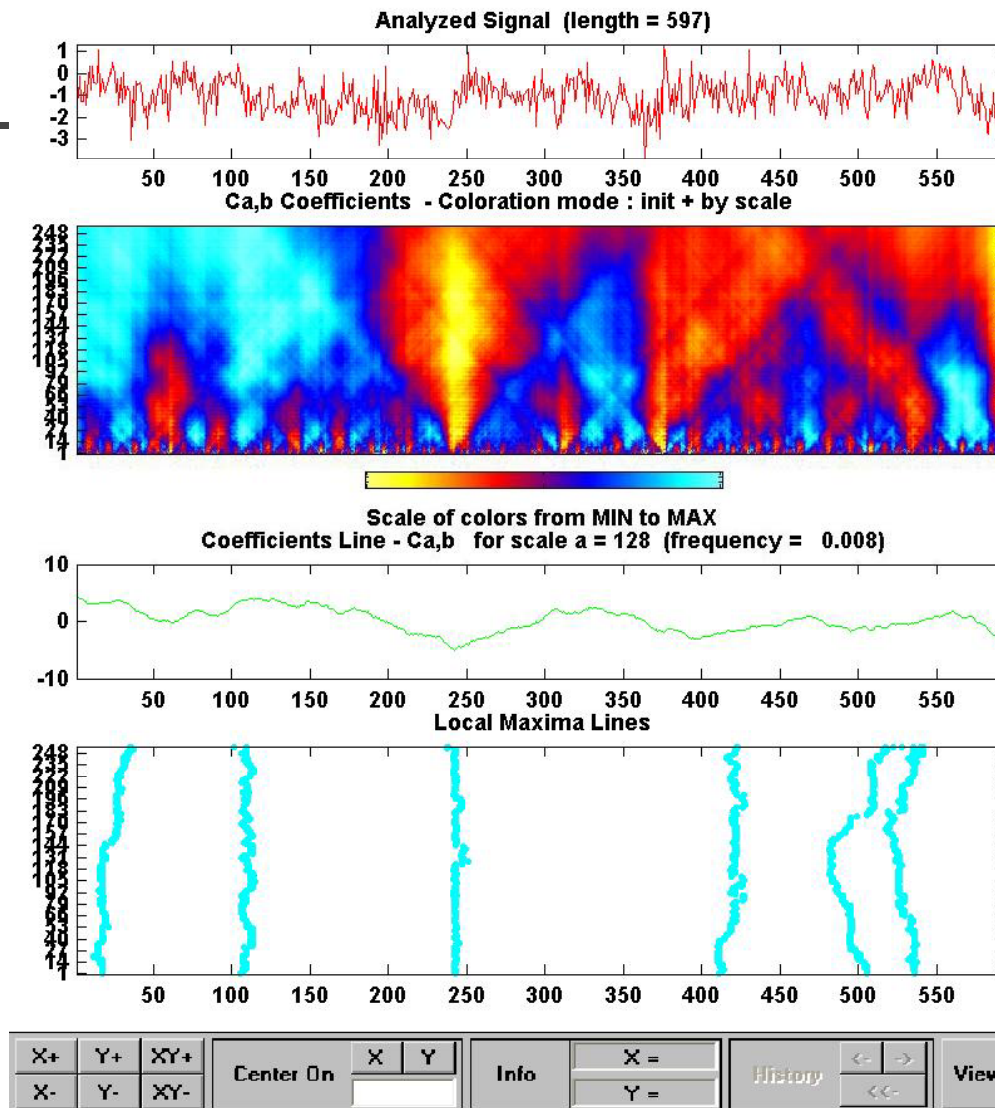
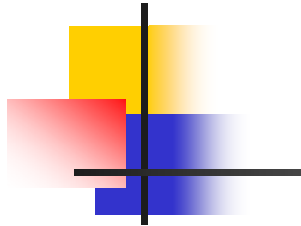


Events data

Date	Actor	Target	Weis Code	Wies event	Goldstein scale
790415	ARB	ISR	223	(MIL ENGAGEMENT)	-10
790415	EGY	AFD	194	(HALT NEGOTIATION)	-3.8
790415	PALPL	ISR	223	(MIL ENGAGEMENT)	-10
790415	UNK	ISR	223	(MIL ENGAGEMENT)	-10
790415	ISR	EGY	31	(MEET)	1
790415	EGY	ISR	31	(MEET)	1
790415	ISRMIL	PAL	223	(MIL ENGAGEMENT)	-10
790415	PALPL	JOR	223	(MIL ENGAGEMENT)	-10
790415	EGY	AFD	193	(CUT AID)	-5.6
790415	IRQ	EGY	31	(MEET)	1
790415	EGY	IRQ	31	(MEET)	1
790415	ARB	CHR	223	(MIL ENGAGEMENT)	-10
790415	JOR	AUS	32	(VISIT)	1.9
790415	UGA	CHR	32	(VISIT)	1.9
790415	ISRGOV	ISRSET	54	(ASSURE)	2.8

177,336 events from April 1979 to October 2003 in Levant data set (KEDS).

Analyzing conflict



Data (Size) **newlevant [597]**

Wavelet **db** **1**

Sampling Period: **1**

Scale Settings

Step by Step Mode

Min (> 0) **1**

Step (> 0) **1**

Max (<= 256) **256**

Analyze

New Coefficients Line

Refresh Maxima Lines

Selected Axes

☒ Coefficients

☒ Coefficients Line

☒ Maxima Lines

☒ Scales ☐ Frequencies

Coloration Mode

init + by scale

Colormap **1 - jet**

Nb. Colors **128**

Brightness **-** **+**

Close



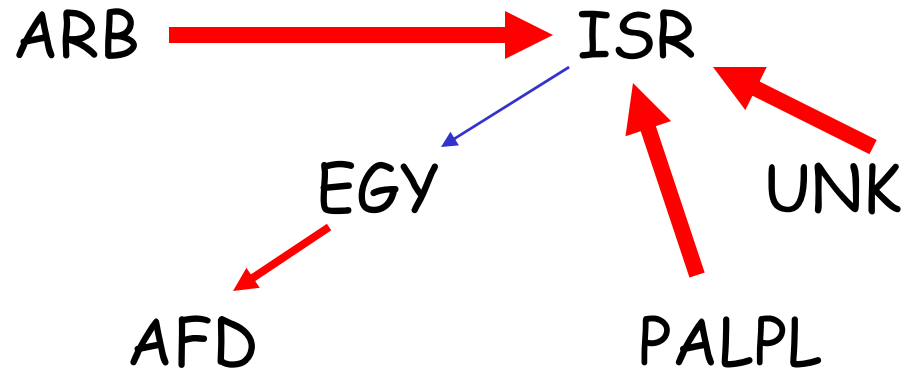
Singularities = MID start/end

biweek	Date range	event
17-35	11/79 to 8/80	Start of Iran/Iraq war
105-111	4/83 to 7/83	Beirut suicide attack
244	1/91 to 2/91	Desert Storm
413-425	1/95 to 7/95	Rabin assassination/start of Intifada
483-518	10/97 to 2/99	US/Iraq confrontation via Richard Butler/arms inspectors
522-539	4/99 to 11/99	Second intifada Israel/Palestine

Key representational idea

- Model interactions between countries in a directed graph.

Date	Actor	Target	Weis Code	Wies event	Goldstein scale
790415	ARB	ISR	223	(MIL ENGAGEMENT)	-10
790415	EGY	AFD	194	(HALT NEGOTIATION)	-3.8
790415	PALPL	ISR	223	(MIL ENGAGEMENT)	-10
790415	UNK	ISR	223	(MIL ENGAGEMENT)	-10
790415	ISR	EGY	31	(MEET)	1

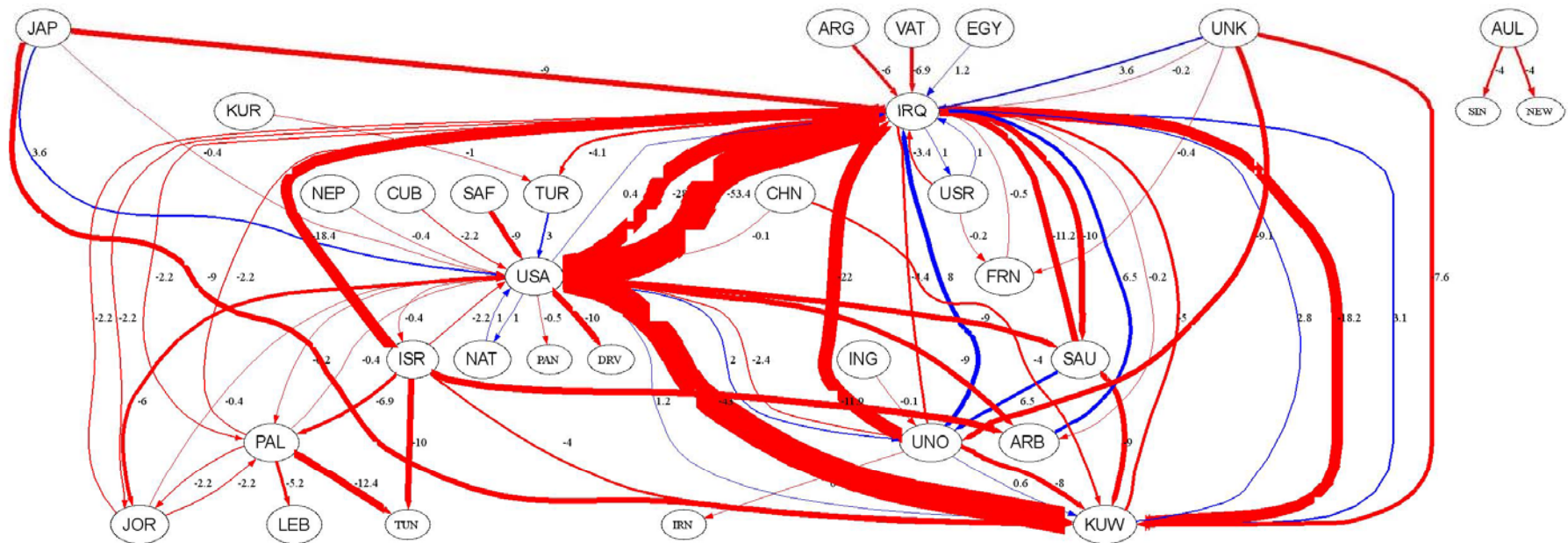




Hubs and authorities for events data

- A hub node is an important initiator of events.
- An authority node is an important target of events.
- Hypothesis:
 - Identifying hubs and authorities over a particular temporal chunk of events data tells us who the key actors and targets are.
 - Changes in the number and size of connected components in the interaction graph signal potential outbreak of conflict.

Hub/authorities for Desert Storm





Validation using MID data

- Number of bi-weeks with MIDS in Levant data: 41 out of 589.
- Result 1: Hubs and Authorities clearly identify actors and targets in impending conflict.
- Result 2: Change in component hubs and authorities scores, change in number of connected components, change in size of largest component 3 weeks before MID, characteristic of MID onset.
- Problem: High false alarm rate. Currently adding substantive political knowledge to rule them out.



Publications

- An OKAPI-based approach for article filtering, Lee, Than, Stoll, Subramanian, 2006 Rice University Technical Report.
- Hubs, authorities and networks: predicting conflict using events data, R. Stoll and D. Subramanian, International Studies Association, 2006 (invited paper).
- Events, patterns and analysis, D. Subramanian and R. Stoll, in Programming for Peace: Computer-aided methods for international conflict resolution and prevention, 2006, Springer Verlag, R. Trappl (ed).
- Four Way Street? Saudi Arabia's Behavior among the superpowers, 1966-1999, R. Stoll and D. Subramanian, James A Baker III Institute for Public Policy Series, 2004.
- Events, patterns and analysis: forecasting conflict in the 21st century, R. Stoll and D. Subramanian, Proceedings of the National Conference on Digital Government Research, 2004.
- Forecasting international conflict in the 21st century, D. Subramanian and R. Stoll, in Proc. of the Symposium on Computer-aided methods for international conflict resolution, 2002.



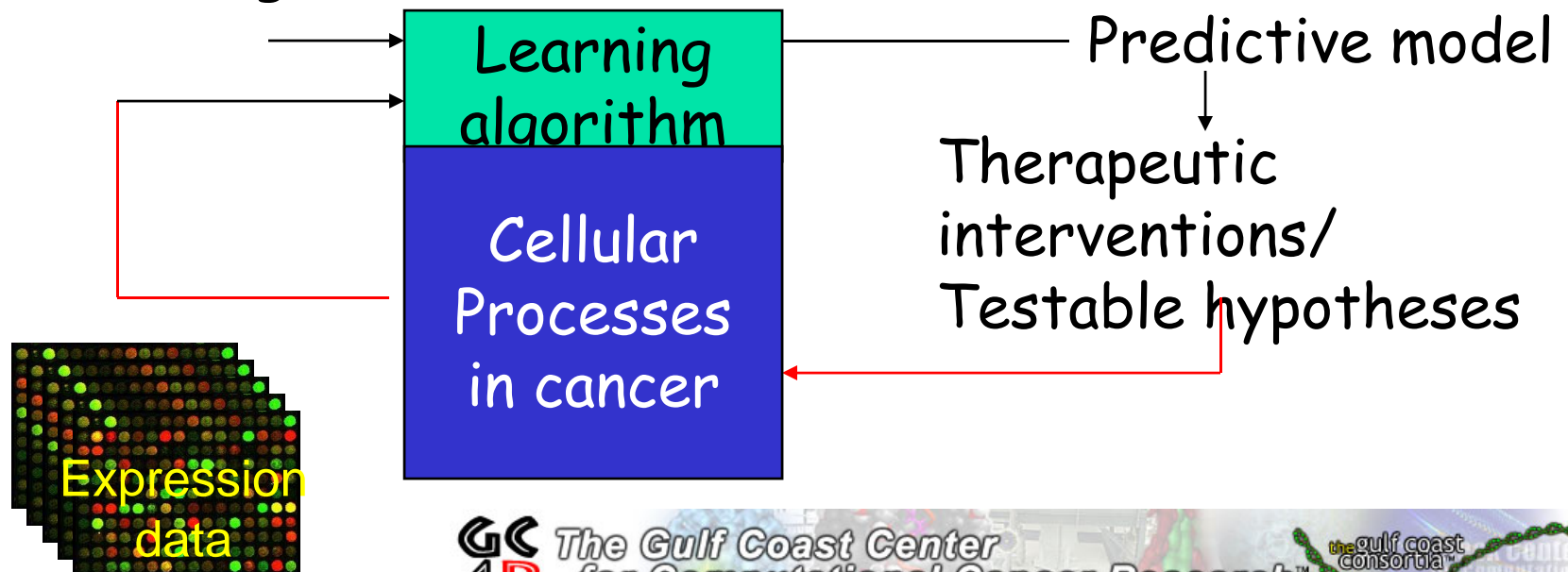
Roadmap of talk

- Four case-studies
 - Unknown system, changing dynamics
 - Tracking human learning on a complex visual-motor task.
 - Predicting the evolution of international conflict.
 - Unknown system, customization needed
 - Learning disruptions in metabolic processes in cancer cells.
 - Designing customized compiler optimization sequences for application programs.

Learning models of cellular process from data

Extract cellular process models
in diseased cells

Prior knowledge

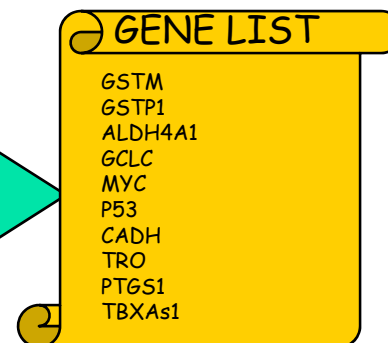
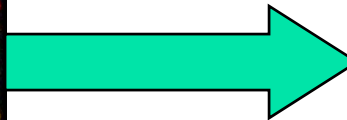
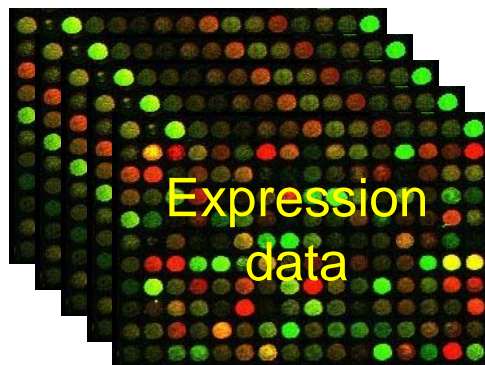


The Gulf Coast Center
for Computational Cancer Research™



Task question

- Can we use gene expression data to identify disrupted cellular processes in diseased cells?



Top 100
up and
down
regulated
genes



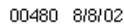
Challenges

- Ab initio learning of cellular process is not possible - data is extremely limited (few hundred samples).
- Data is noisy; measurement and interpretation problems, as well as problems caused by tissue heterogeneity.
- Therefore, we need to incorporate available knowledge of biological processes; the role of expression data is to refine known models.



Approach

- Represent components of cellular processes as Bayesian networks.
- Learn parameters of component networks by expectation maximization.
- Compose component networks.
- Results: glutathione and urea metabolism in prostate cancer



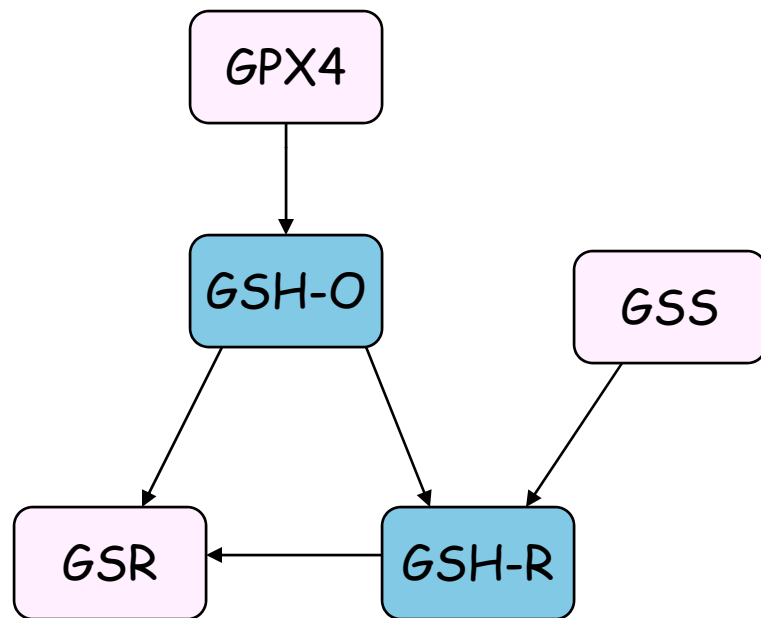
An important metabolic process; detoxification in cells. Known to be disrupted in several cancers.



Modeling cellular processes

- We represent portions of metabolic pathways as Bayesian networks.
 - Friedman 1999, Friedman 2000, Segal 2003.
- Nodes denote the expression levels of genes **as well as gene products**. Expression levels are discretized into three categories (low, medium, high).
 - Most Bayesian network models in the literature only represent observable gene expression levels.
- Edges denote producer-consumer and catalyst relationships between nodes.
 - Very little in the literature on how to model these relationships so that the network is learnable from available data.

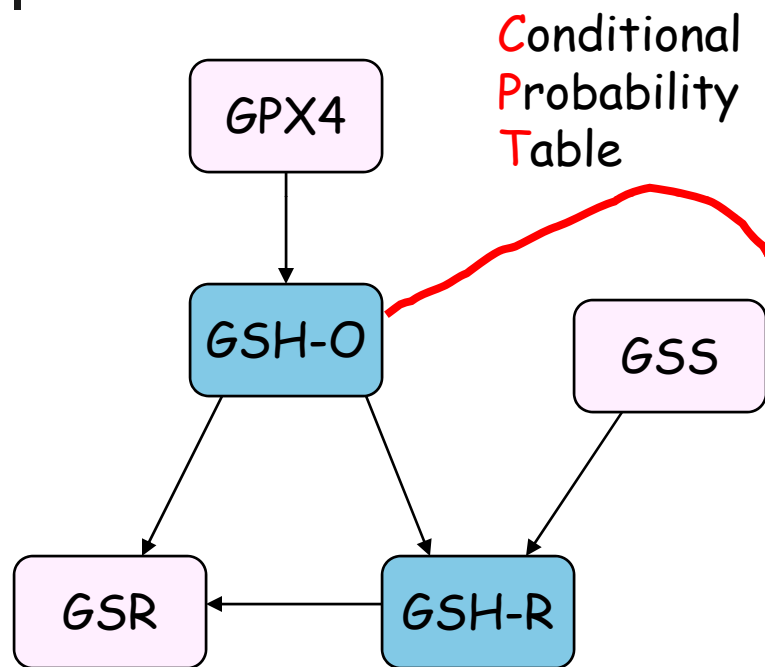
Modeling cellular processes: topology of network



A portion of the GSH network

- Three alternate synthesis pathways for GSH-R: from GSH-O by GSR, from GSH-O by GPX4, and independently from GSS.
- Edges here are not causal; edge directions chosen to
 - Keep network acyclic
 - Make nodes have no more than two to three parents.
- Network is an alternate but correct factoring of the full joint distribution on expression levels.

Modeling cellular processes: the quantitative parameters



A portion of the GSH network

- Our models have a quantitative component. Each node has a conditional probability distribution associated with it.
- These models are learned from data!

GPX	GSH-O (normal)		
	low	med	high
low	0.67 ± 0.25	0.23 ± 0.24	0.10 ± 0.24
med	0.33 ± 0.40	0.65 ± 0.40	0.00 ± 0.01
high	0.04 ± 0.07	0.13 ± 0.10	0.83 ± 0.09

GPX	GSH-O (tumor)		
	low	med	high
low	0.74 ± 0.35	0.11 ± 0.16	0.14 ± 0.32
med	0.68 ± 0.34	0.09 ± 0.13	0.23 ± 0.27
high	0.02 ± 0.02	0.02 ± 0.02	0.96 ± 0.02



Component network learning

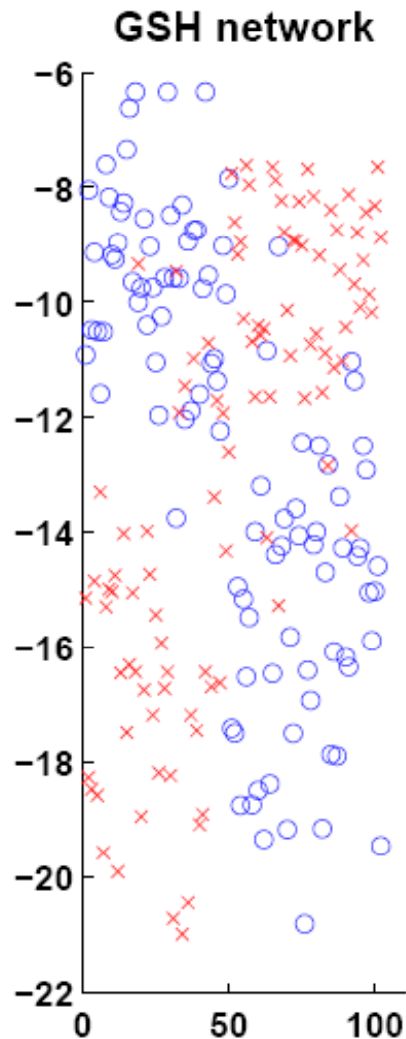
GPX	GSH-O (normal)		
	low	med	high
low	0.67 ± 0.25	0.23 ± 0.24	0.10 ± 0.24
med	0.33 ± 0.40	0.65 ± 0.40	0.00 ± 0.01
high	0.04 ± 0.07	0.13 ± 0.10	0.83 ± 0.09

GPX	GSH-O (tumor)		
	low	med	high
low	0.74 ± 0.35	0.11 ± 0.16	0.14 ± 0.32
med	0.68 ± 0.34	0.09 ± 0.13	0.23 ± 0.27
high	0.02 ± 0.02	0.02 ± 0.02	0.96 ± 0.02

Note that tumor cells produce lower than normal amounts of GSH-O when GPX levels are medium.

- We learn **separate network** parameters for normal cells and diseased cells for each metabolic process we model.
- Differences in parameters indicate differences in the underlying process.

Evaluating EM learning



- Samples 1-50 are normal, 51-102 are tumor.
- Blue circles denote log-likelihood of sample computed from the learned normal GSH network, red crosses denote log-likelihood of sample computed from the learned tumor GSH network.
- Note the nice separation of normal and tumor samples! The EM procedure finds good parameter values that characterize the GSH process in normal and tumor cells.

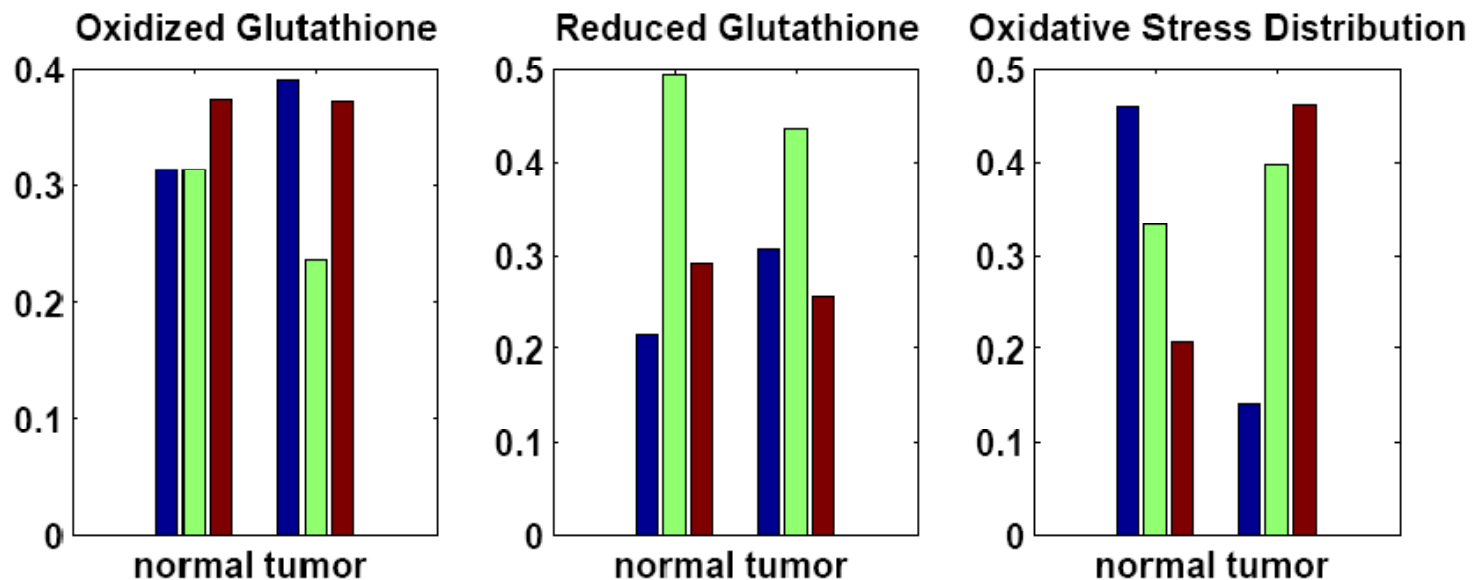


Robustness of EM learning

Leave-one-out Cross validation results for the *GSH* network

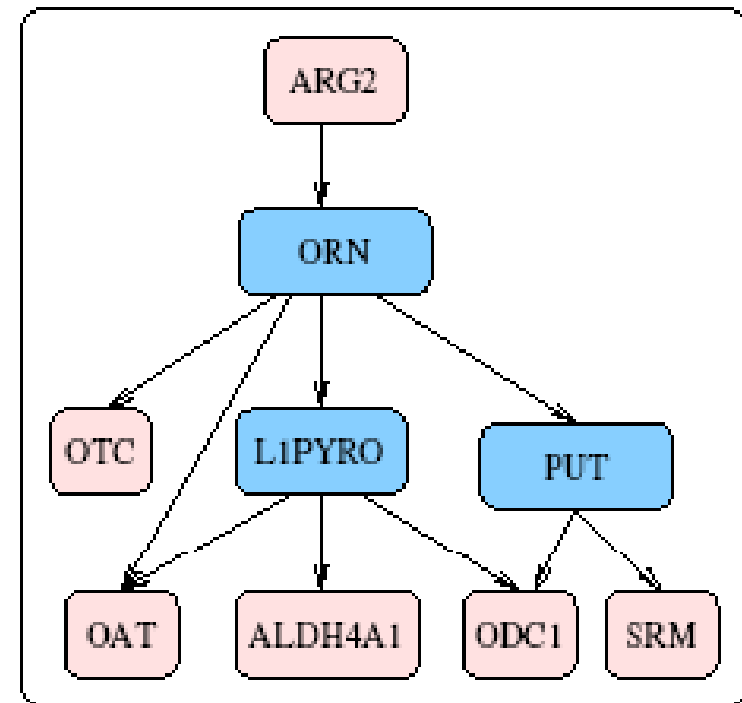
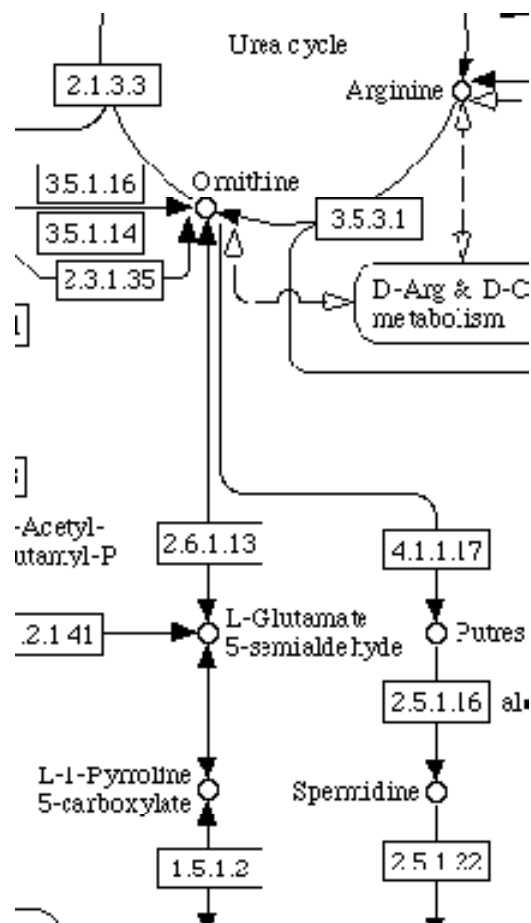
	GSH Network	
Predicted	Actual	
	N	T
N	41	8
T	9	44

Predictions from GSH network

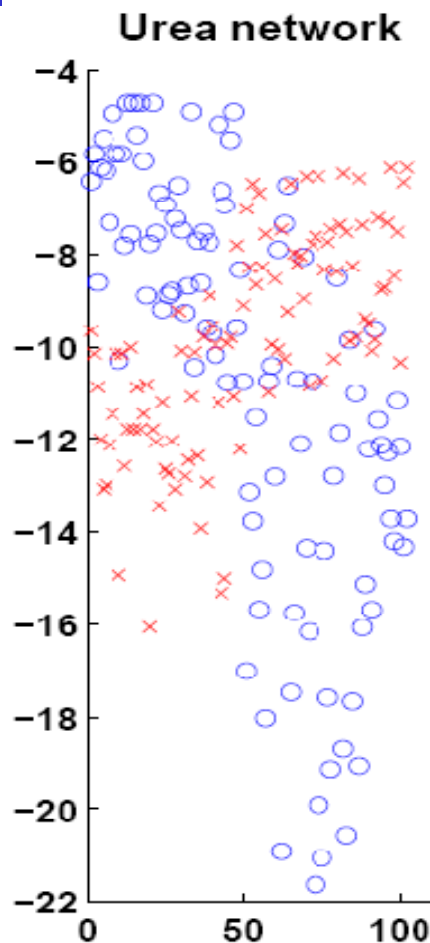


We can make predictions about metabolite levels from the two learned networks. It is remarkable that **we can predict** that the **level of oxidative stress in tumor cells is much higher** in tumor cells using networks learned **from the gene expression data** alone!

The urea network



Evaluating EM learning



- Samples 1-50 are normal, 51-102 are tumor.
- Blue circles denote log-likelihood of sample computed from the learned normal urea network, red crosses denote log-likelihood of sample computed from the learned tumor urea network.
- Note the nice separation of normal and tumor samples! The EM procedure finds good parameter values that characterize the urea process in normal and tumor cells.

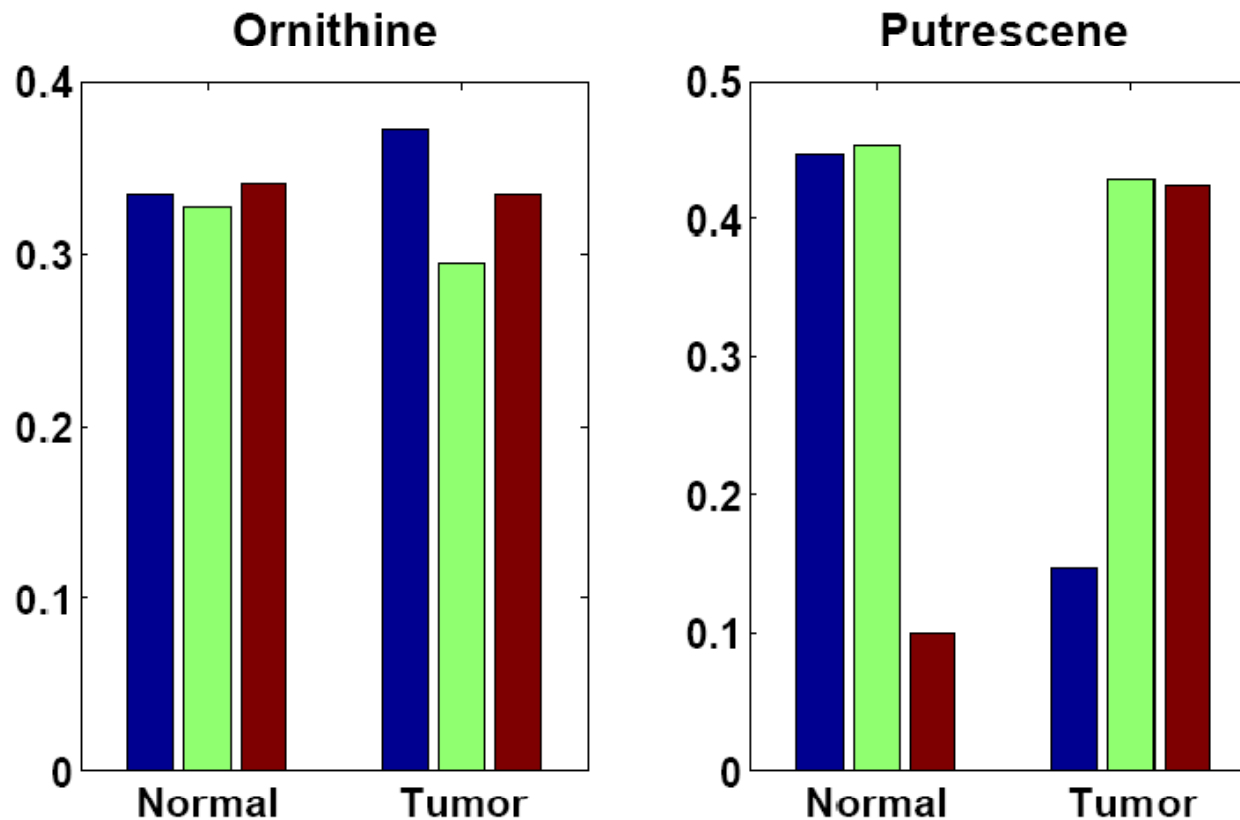


Robustness of EM learning

Leave-one-out Cross validation results for the urea network

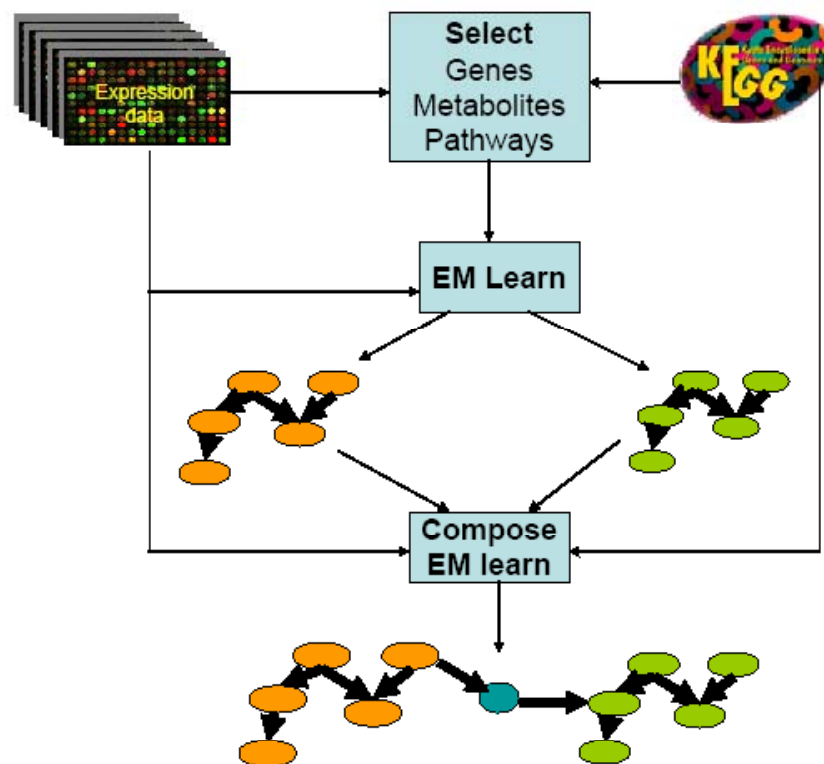
	GSH Network		Urea Network	
Predicted	Actual		Actual	
	N	T	N	T
N	41	8	42	13
T	9	44	8	39

Predictions of the urea network



The metabolite Putrescine is overexpressed in the urine of prostate cancer patients: a clinical finding, which we can derive from gene expression data!

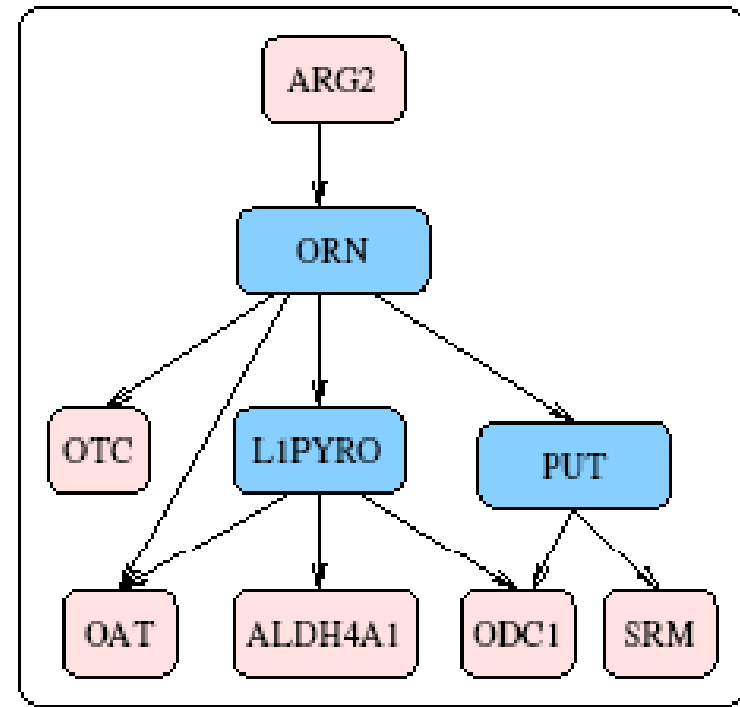
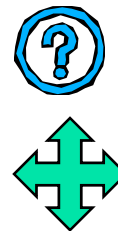
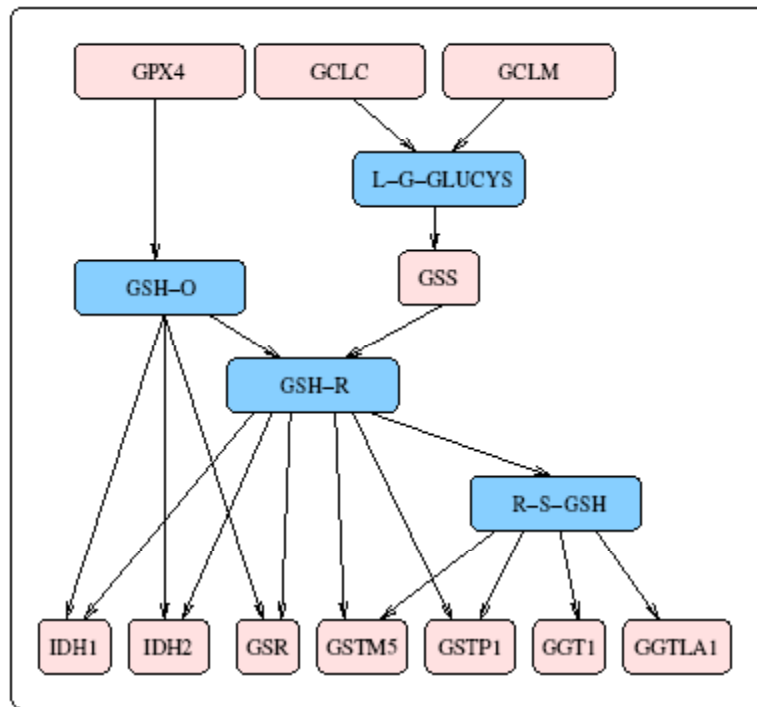
PAPES: scaling up the learning



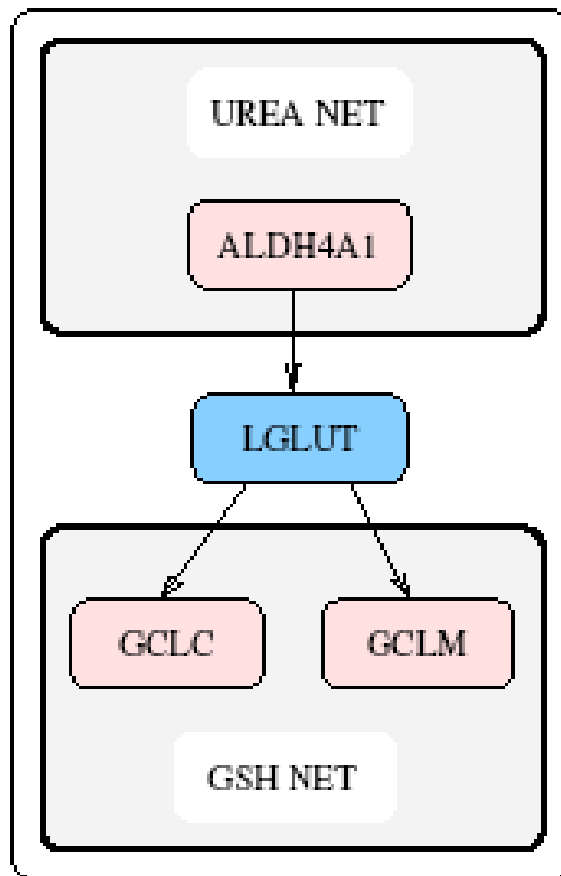
Learn component networks

Compose component networks

Composing networks



The composed network



- Find genes/metabolites in a network that mediate genes/metabolites in another network, either directly or indirectly via a new gene/metabolite.

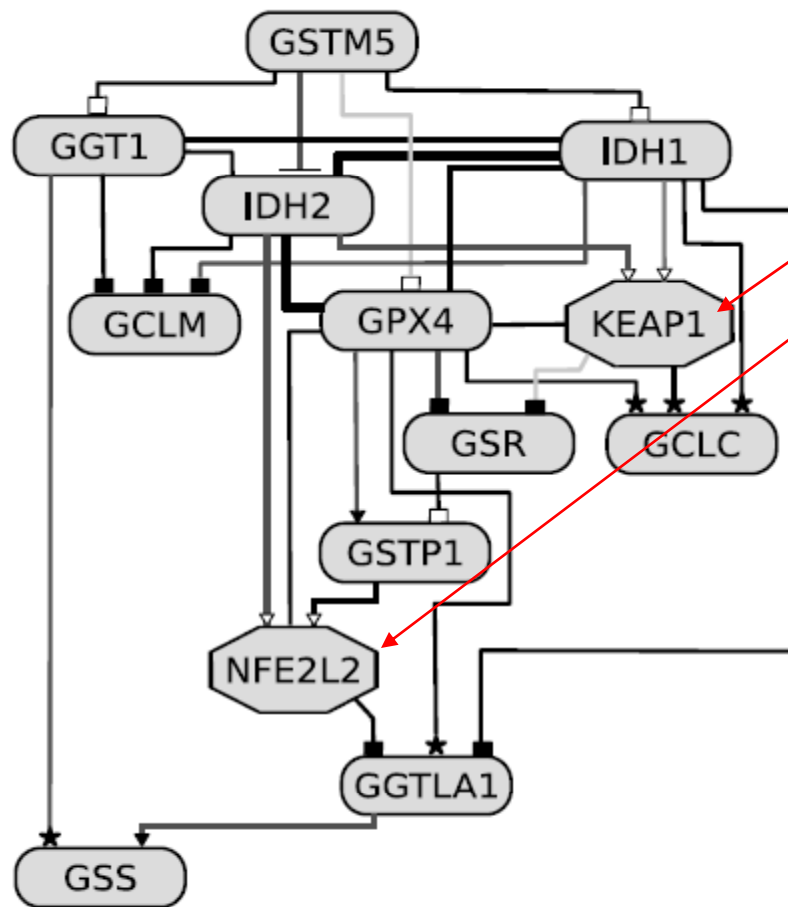


How well does the composition work?

Leave-one-out Cross validation results for the combined network

	GSH Network		Urea Network		Combined Network	
Predicted	Actual		Actual		Actual	
	N	T	N	T	N	T
N	41	8	42	13	45	7
T	9	44	8	39	5	45

Learning regulatory networks



Learn structure of network with putative transcription factors.

The right transcription factors improve likelihood scores dramatically!

Transcription factors appear to attach themselves to the Markov blankets of the enzymes they regulate.



Summary

- We can learn component networks representing portions of cellular processes from gene expression data.
- By learning separate networks for normal and diseased cells, we can determine which cellular process are potentially compromised in diseased cells.
- We can compose component networks to get more accurate explanations of the differences between normal and diseased cells.
- We can discover transcription factors and their roles using structure learning.



Publications

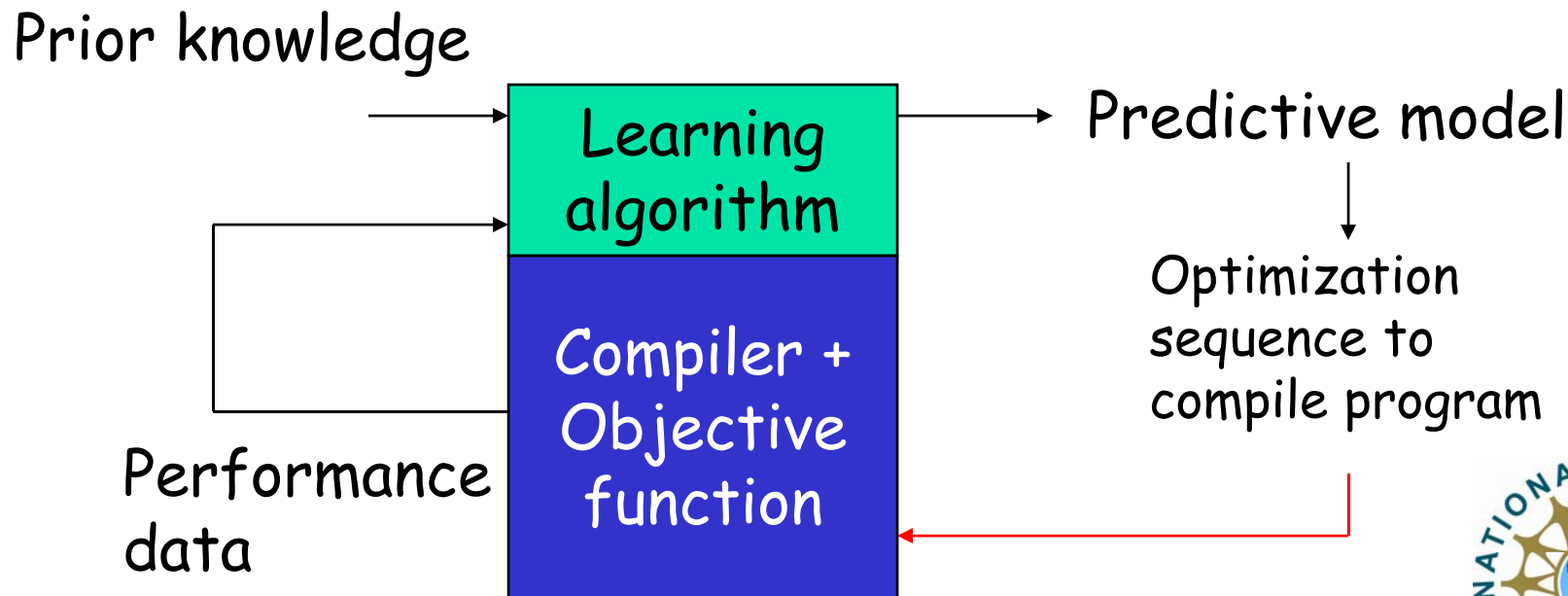
- Computational methods for learning bayesian networks from high-throughput biological data, B. M. Broom and D. Subramanian, in edited by K.A. Do and P. M. Mueller and M. Vanucci, Cambridge University Press, 2006.
- Predicting altered pathways using extendable scaffolds, B. M. Broom, T. J. McDonnell and D. Subramanian, in International Journal of Bioinformatics Research and Applications, 2006.
- Predicting altered pathways using extendable scaffolds, B. M. Broom, T. J. McDonnell and D. Subramanian, in BIOT 2005.



Roadmap of talk

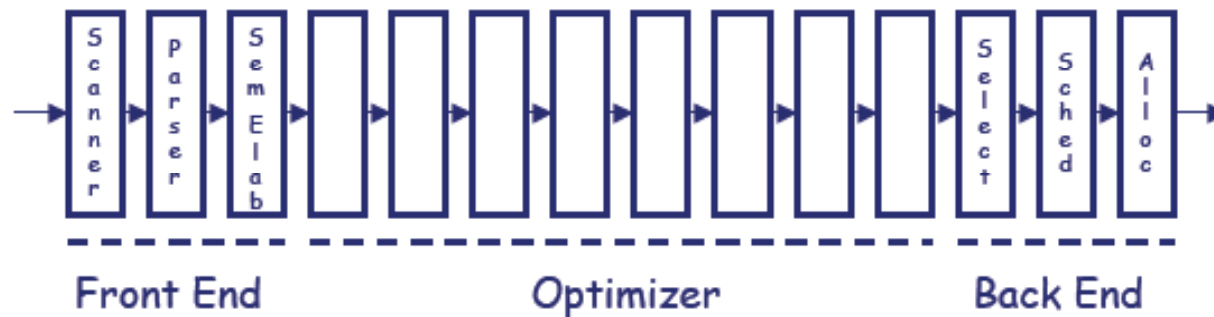
- Four case-studies
 - Unknown system, changing dynamics
 - Tracking human learning on a complex visual-motor task.
 - Predicting the evolution of international conflict.
 - Unknown system, customization needed
 - Learning disruptions in metabolic processes in cancer cells.
 - Designing customized compiler optimization sequences for application programs.

Customizing optimization sequences for compilers



Compilers 101

Compilers are well understood



What order should the optimizations run in?
Classic answer: compiler writer determines chooses
a universal sequence at design time.



Compiler optimizations

- Code Transformations

g, l, u, v, x, y, z: redundancy elimination methods

s: register coalescing

n: useless control flow elimination

c: constant propagation

d: dead code elimination

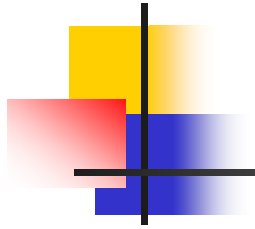
o: peephole optimization

p: iteration peeling

r: algebraic re-association

t: operator strength reduction

m: renaming



Why is problem difficult?

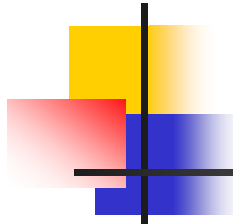
- Many optimizations available (16 in our compiler, and a hundred in the literature).
- “Compiler optimizations are like poetry: more are written than are published in commercial compilers”.
- Interactions between optimizations not well understood.
 - Difficult to analytically predict the impact of a optimization sequence on a program.
- Optimization sequences affect different programs differently.



Why compilers need to learn

- We do not know enough, **today**, to predict a good optimization sequence.
- Our prototype system samples the space of sequences and learns its characteristics.
- It uses learned models to predict good sequences for related programs.
- Three Rice PhDs theses on this paradigm
 - Phil Schielke, 2000
 - Alex Grosul, 2004
 - Yi Guo, 2007





Benchmarks

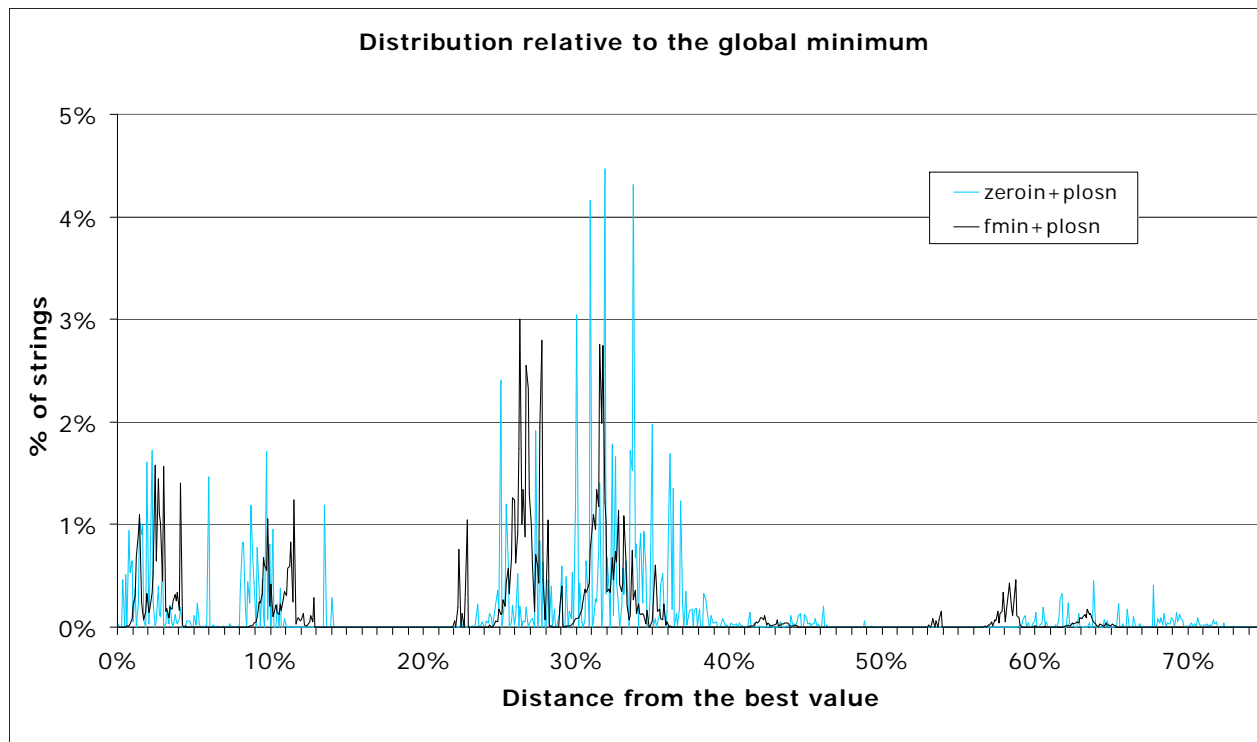
- fmin and zeroin
 - **Exhaustive** enumeration over 5 optimizations with compilation sequences of length 10: 5^{10} sequences.
- To select the best algorithm for searching for a good sequence, need to develop understanding of the sequence space.



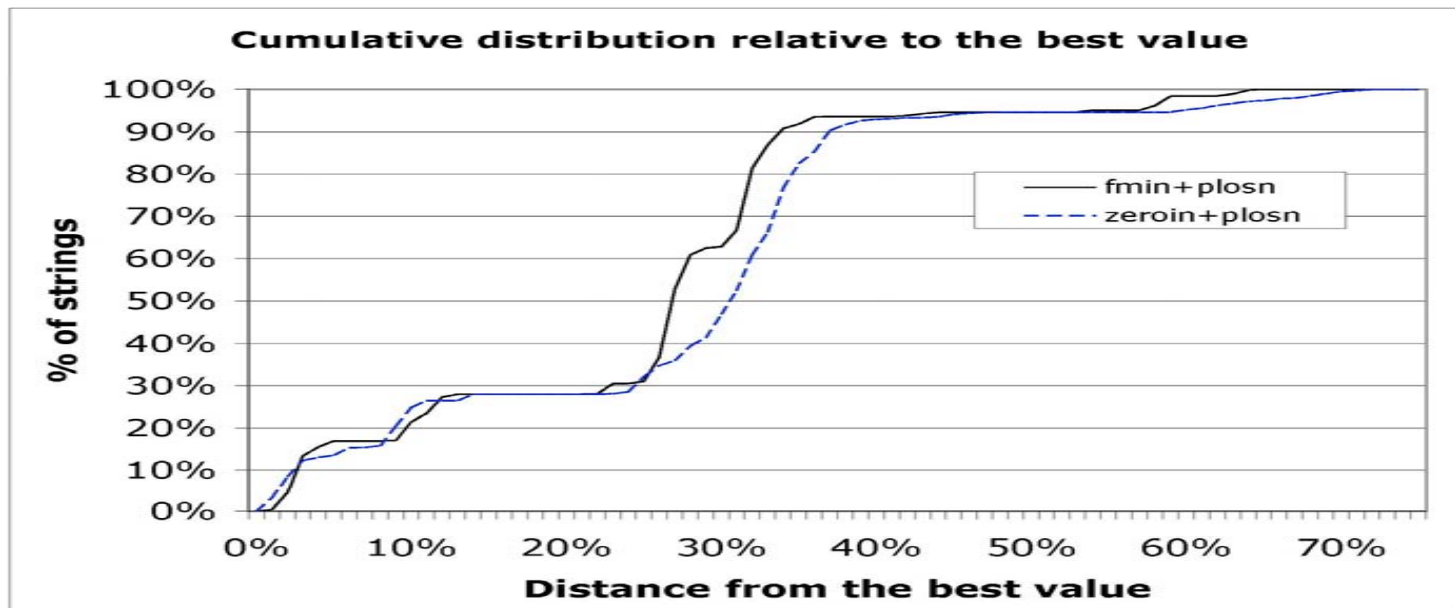
Three questions

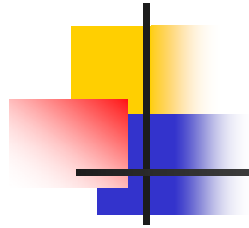
- What percentage of sequences have values within a given percentage of optimal?
- How are these good sequences distributed in the space?
- Are local minima numerous? And how are their values distributed?

Distribution of solutions



Cumulative distribution

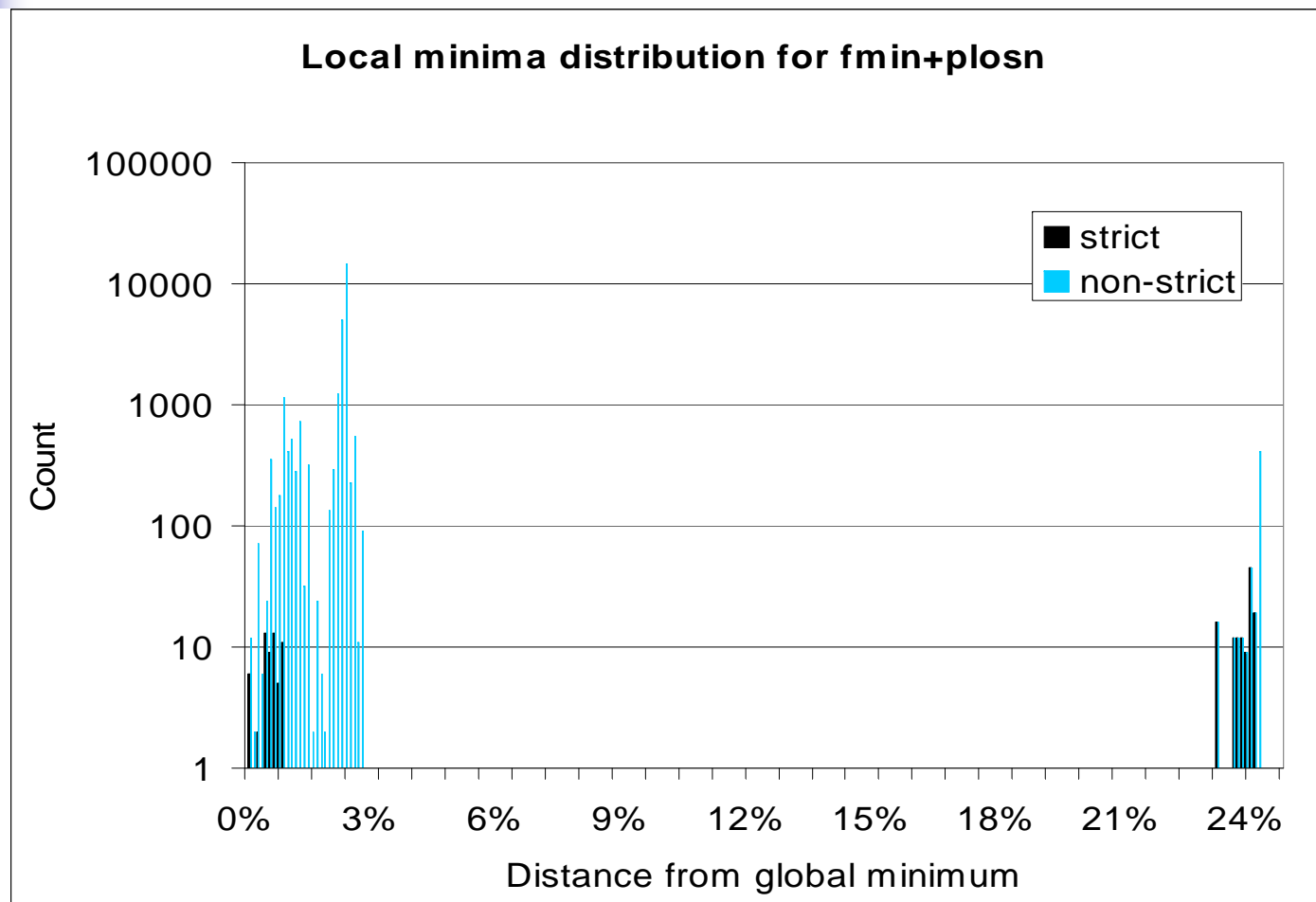




Sequence space for f_{\min}

- Very few solutions (1000 out of 9 million) are within 2.6% of optimal.
- 15% of sequences within 10% of optimal, 30% of sequences within 20% of optimal.

Local minima in sequence space (fmin)

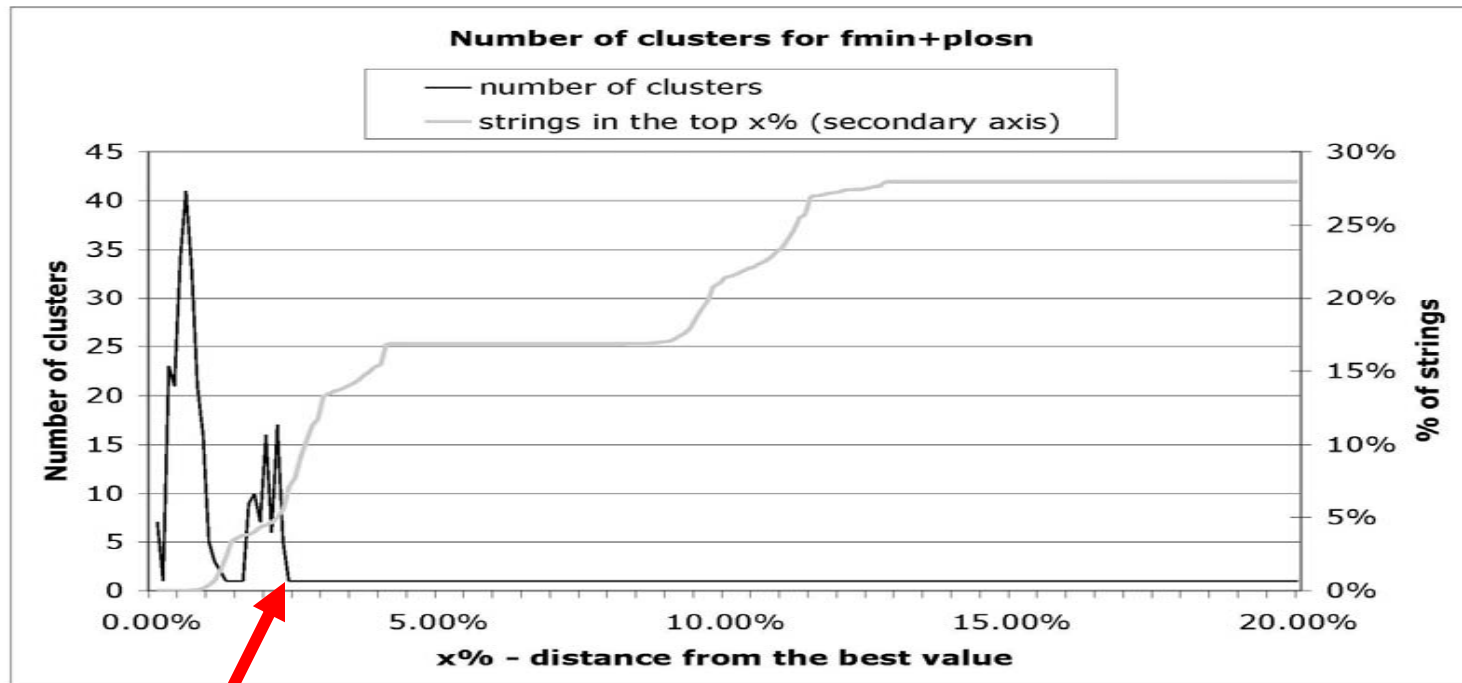




A definition of neighbor

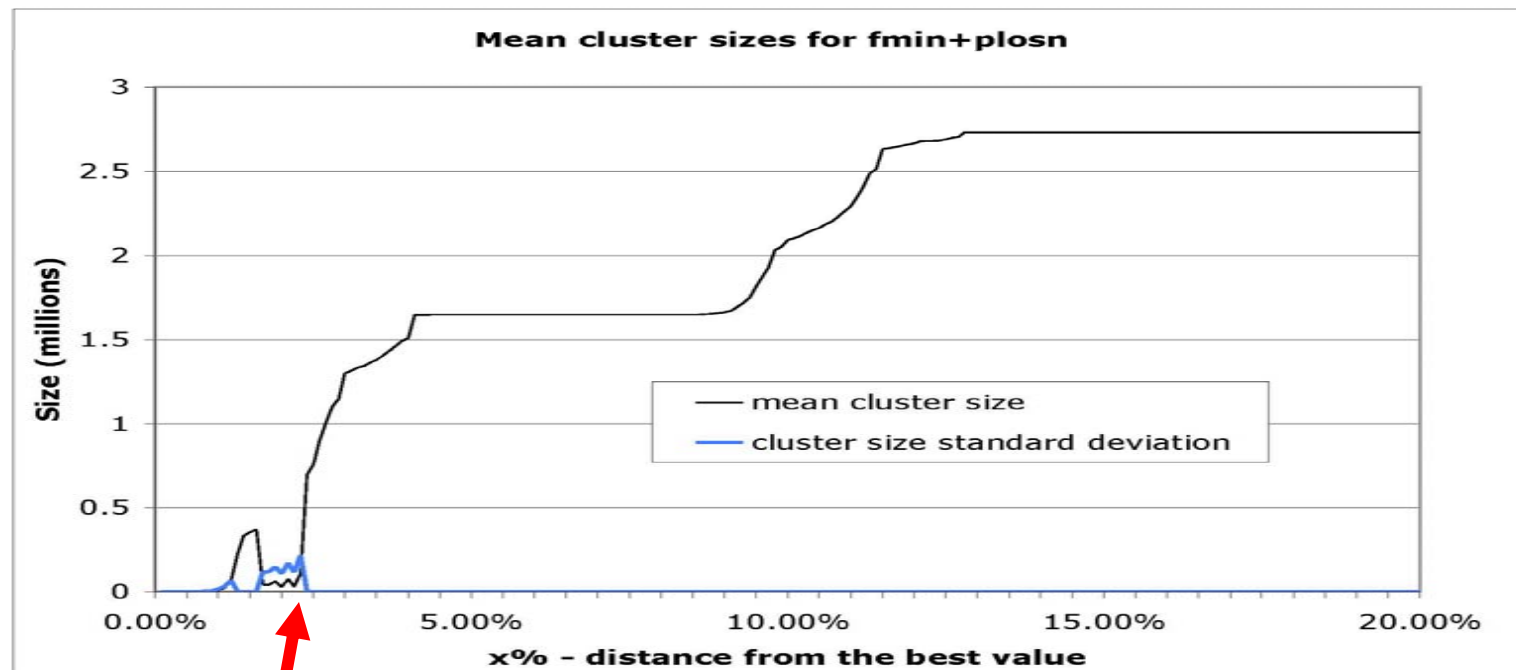
- A sequence s is a neighbor of sequence t , if the Hamming distance between s and t is 1, i.e., the sequences differ in exactly one position.

Number of clusters



Phase transition

Cluster sizes



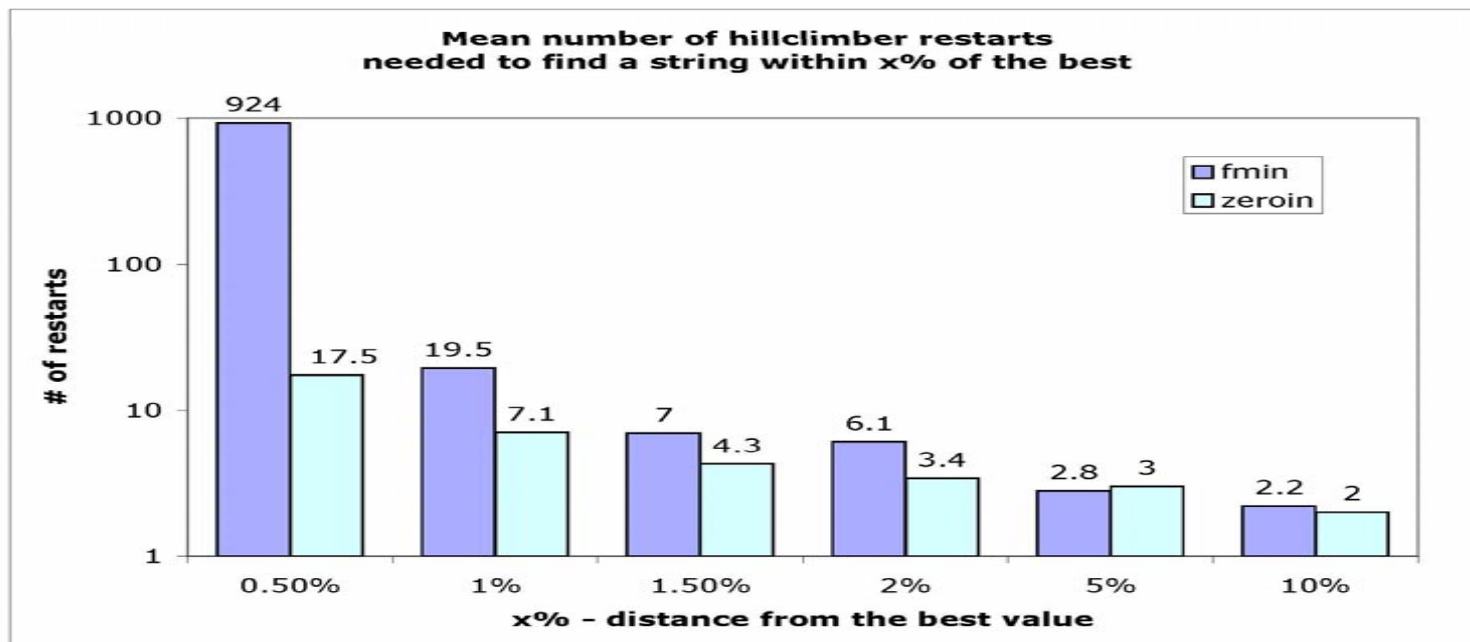
Phase transition



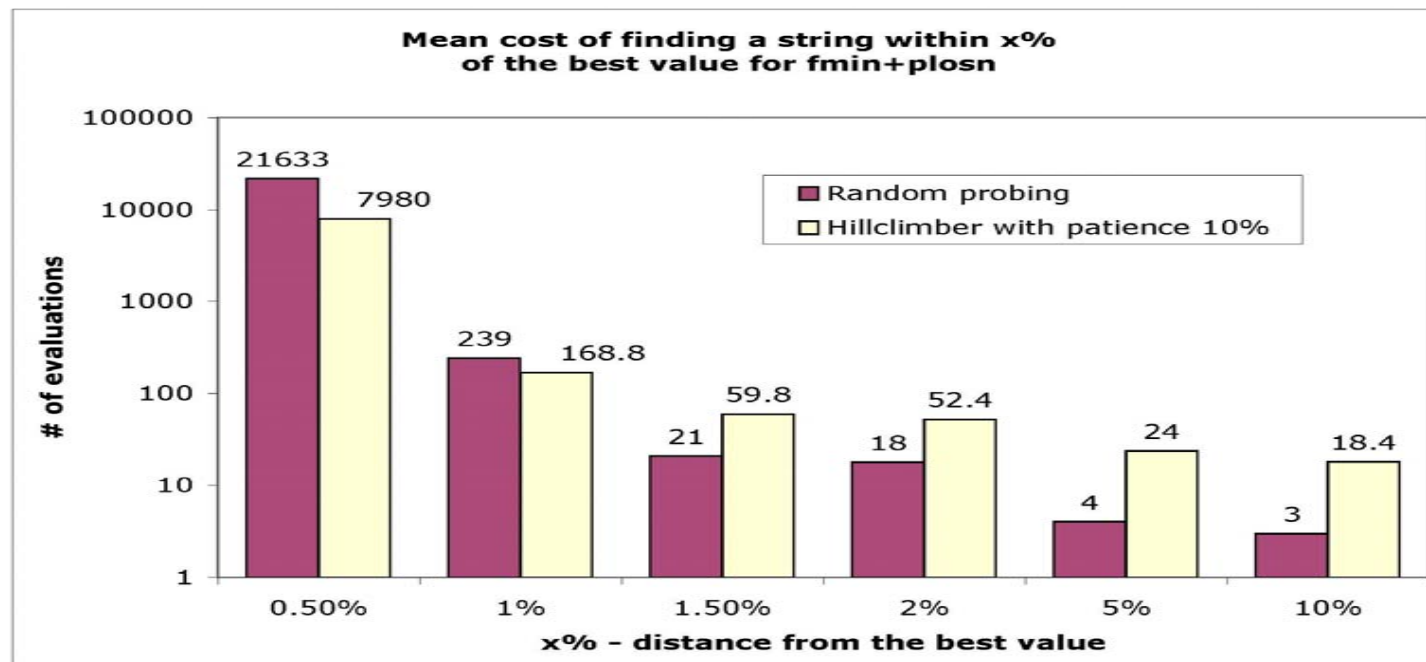
Implication of phase transition

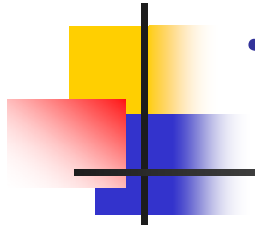
- Finding a solution that is within 2.6% of optimal is qualitatively much harder than finding one $> 2.6\%$ of optimal.
- Expect a hill climber to take many many restarts to find a solution in the $< 2.6\%$ region.

The restart data



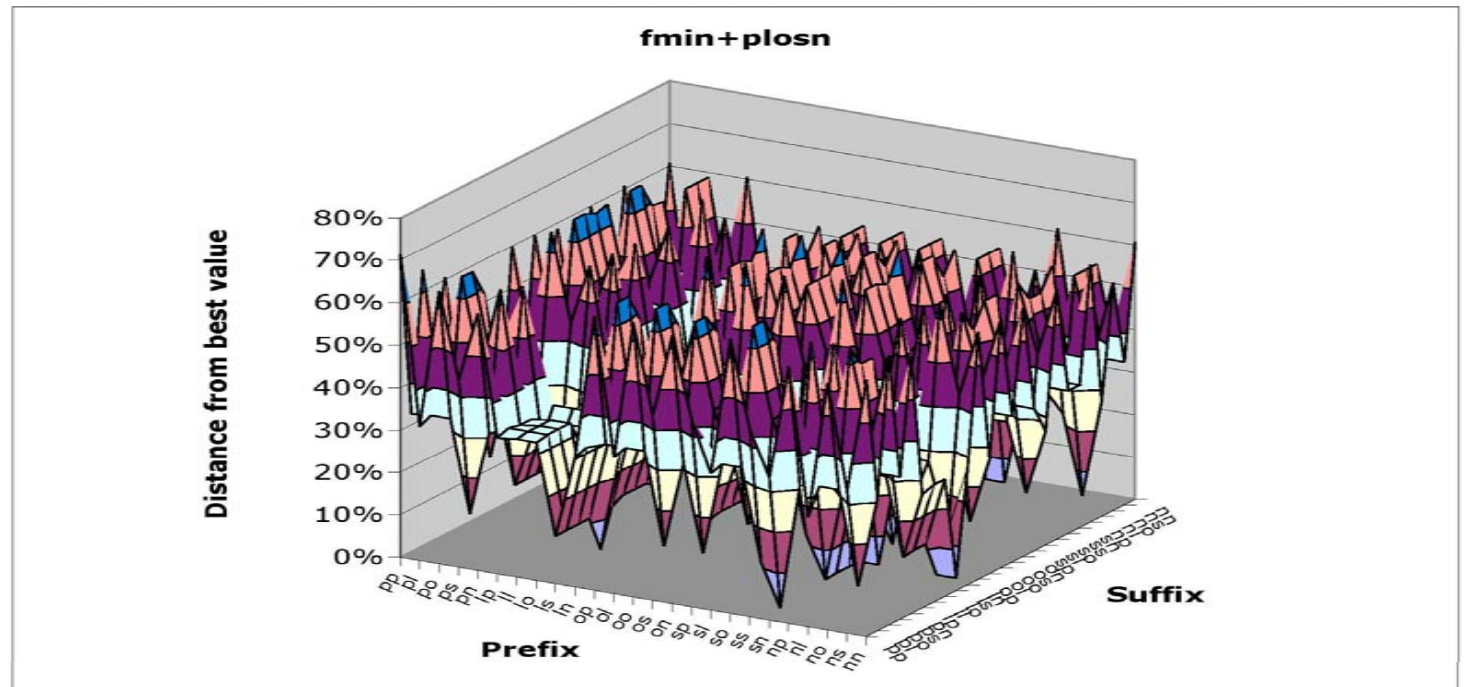
Impatient hill climber vs random probing



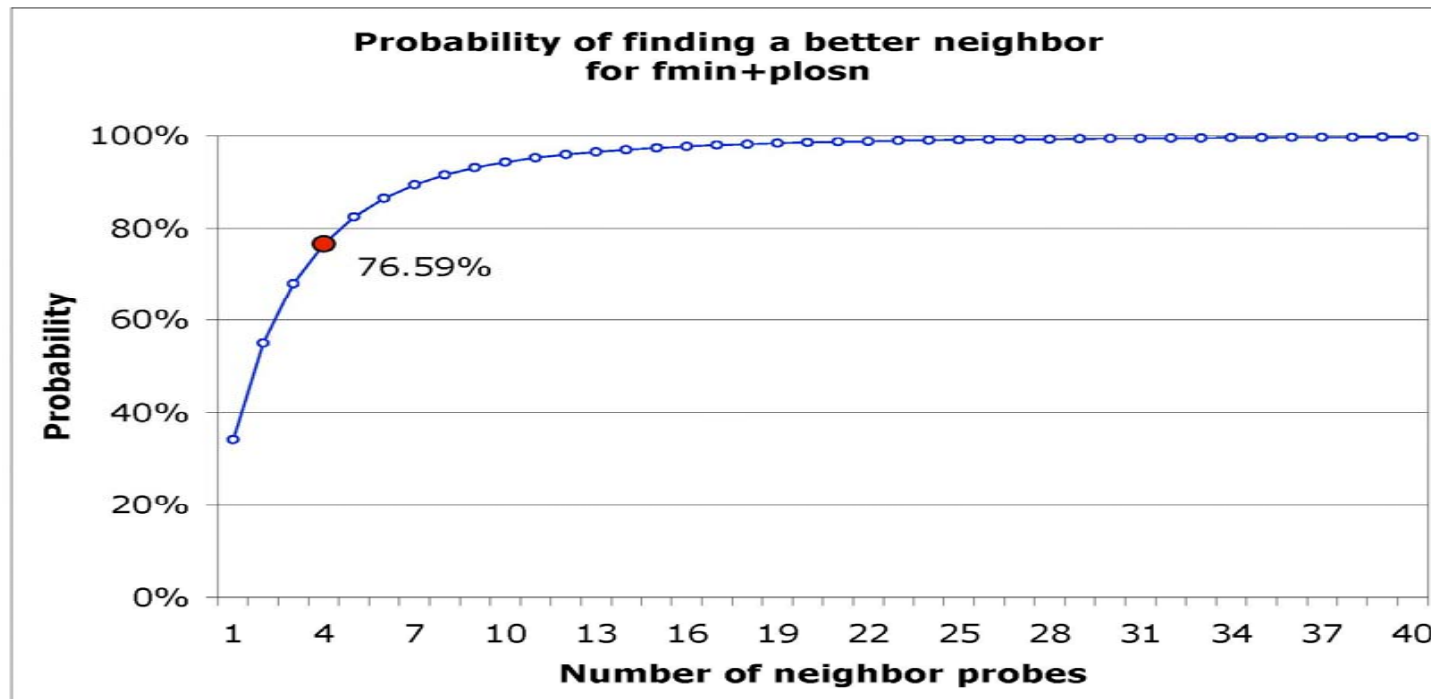


The solution space

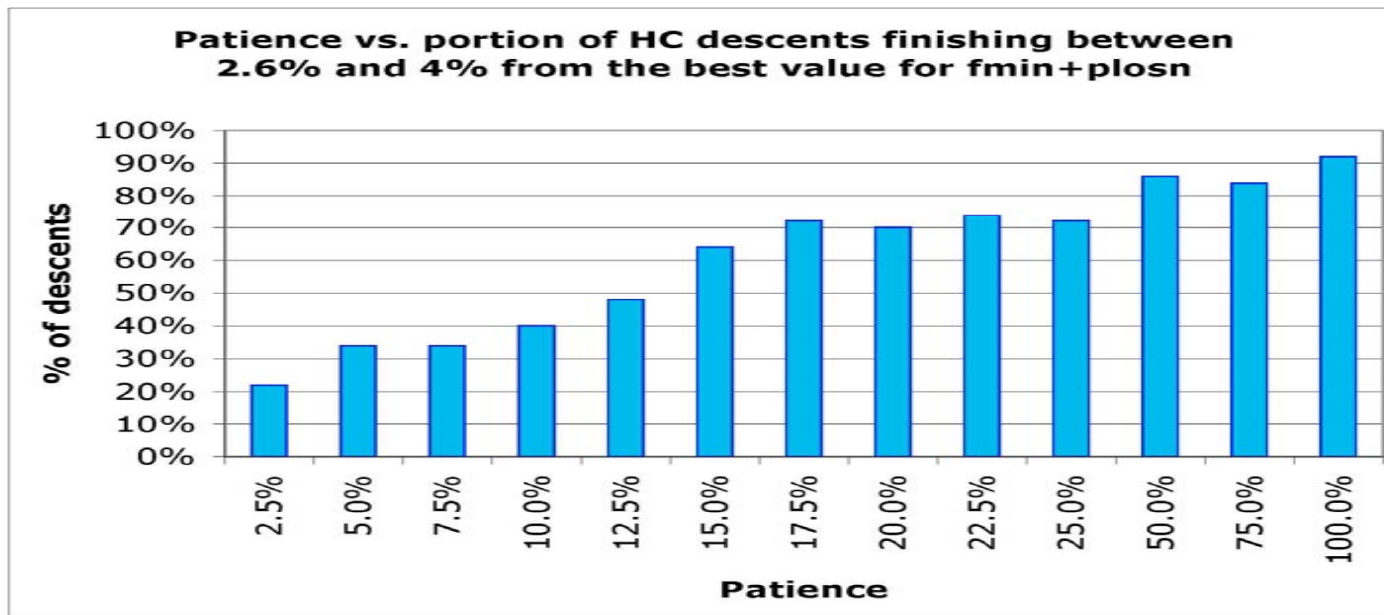
Values of
sequences
of length 4.



Probability of finding a better neighbor



Impatience levels of HC





Summary of terrain properties for fmin

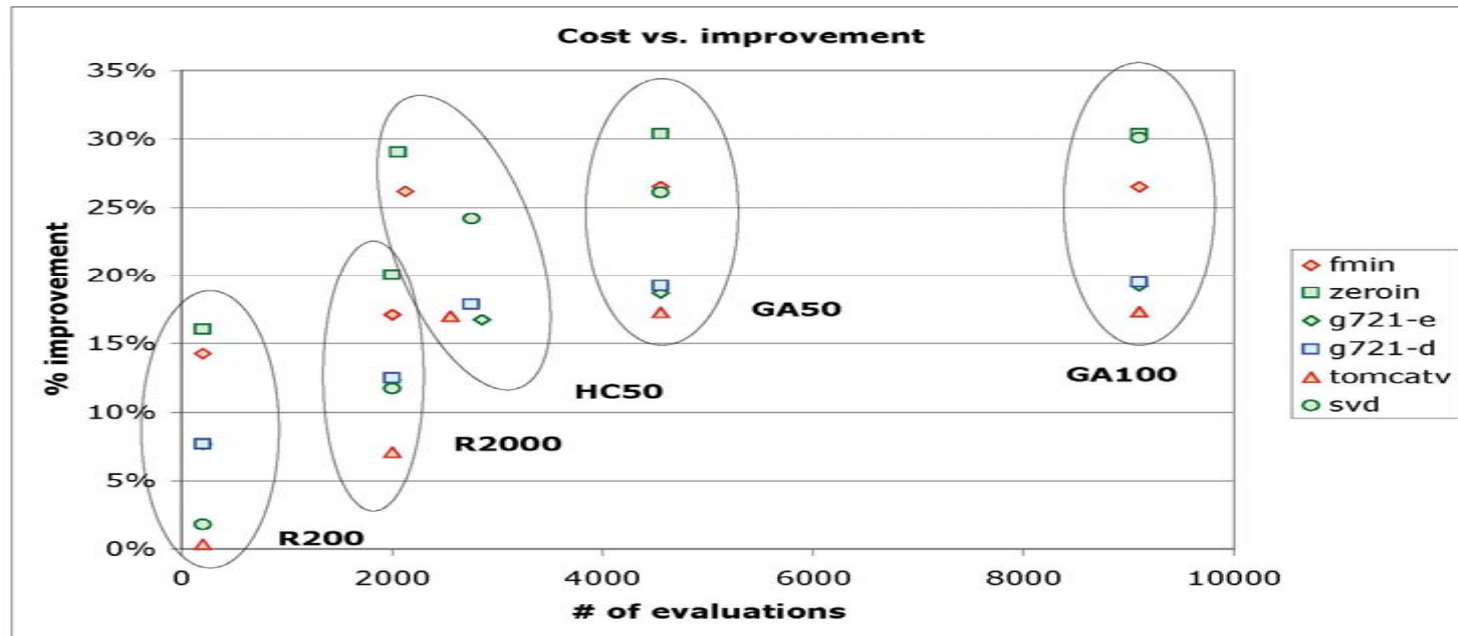
- Many local minima.
- Short downhill runs.
- Starting points matter: cannot reach deep local minima unless one is within 2 steps of it.
- Randomized restarts with impatience (examining 10% of neighbors) is a good algorithmic choice.

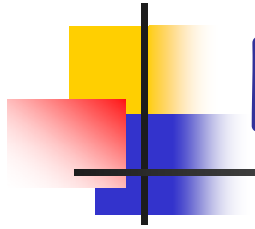


A larger experiment

- Find sequences for SPEC and Media benchmarks by
 - Impatient (10%) randomized restart (50) hill climber
 - GA (pop. size 50 and 100, mutation: 0.02, 1 point crossover, fitness-proportional selection, 10% elitism)

Local search for good sequences





Features - Dormant

- A transformation of a compilation sequence is *dormant* if that transformation has no effect in changing the code. (Hash the signature of the intermediate program)
- Some transformation will become *dormant* if applied after other more aggressive transformations. We say the former transformation is *disabled* by the latter.

c	o	d	c	d	t	p	v	r	x	u	d
---	---	---	---	---	---	---	---	---	---	---	---



Features Selection - Interacting Transformations

We do experiments to find out all *interacting transformations pairs* (a,b) . Such pairs compose our features set F .

- $(a,b) \in F$ if
 - b is never *disabled*, or
 - we can find a program and a sequence s such that
 - $P(s) = P(sb)$, and // b is dormant
 - $P(sab) \neq P(sa)$ // b is not dormant
 - a creates opportunities for b

$P(s)$ is the performance feedback for sequence s .



Features - Count Interacting Occurrences

- Given a sequence, suppose a appears before b in the sequence, (a, b) is an *interacting occurrence* if
 - Both a and b are not *dormant*, and
 - there is no a or b in between them, and
 - b were not *dormant* at any point after a is applied.

c	o	d	c	d	t	p	v	r	x	u	d
---	---	---	---	---	---	---	---	---	---	---	---



Feature Extraction- Example

p p v p x p o c d p d n

pp	pv	px	vp	xp	od	pd	...
4	1	0	1	1	1	1	...

Currently, we have 68 interacting transformations.

Principal Component Analysis shows 16-18 dimensions are enough to get an accuracy over 90%.



Learning Methods

- Learn function mapping feature vector to {good,bad} based on a large training set.
 - Naïve Bayesian Classifier
 - $P(\text{Good} \mid X) > P(\text{Bad} \mid X) \Rightarrow \text{Classified as good}$
 - Regression (Linear and Logistic)
 - Outcome is 1 for Good, 0 for Bad
 - Predicted $Y > 0.5 \Rightarrow \text{Classified as Good}$
 - Otherwise $\Rightarrow \text{Classified as Bad}$
- Function biases next moves in a randomized restart hill climber.



Results: Snapshot after 100 trials

	Fixed	GA	LC-R	LC--L	Best Known
fmin	1815	9.53%	9.20%	10.32%	13.44%
zeroin	1422	5.49%	6.89%	6.12%	9.35%
spline	763	-	-	-	11.14%
si	1544865 6	17.91%	15.30%	22.02%	32.84%
saxpy	9600000	0	0	0	0
sgemv	531000	0	-	-	4.37%
mean		5.49%	5.23%	6.41%	11.86%



Results: Snapshot after 200 trials

	Fixed	GA	LC-R	LC--L	Best Known
fmin	1815	10.63%	13.17%	12.29%	13.44%
zeroin	1422	6.19%	6.89%	7.03%	9.35%
spline	763	1.57%	-	7.99%	11.14%
si	1544865 6	19.03%	18.28%	26.86%	32.84%
saxpy	9600000	0	0	0	0
sgemv	531000	0.15%	-	1.13%	4.37%
mean		6.26%	6.39%	9.22%	11.86%



Results: Snapshot after 500 trials

	Fixed	GA	LC-R	LC--L	Best Known
fmin	1815	12.62%	13.33%	13.33%	13.44%
zeroin	1422	6.89%	7.03%	9.14%	9.35%
spline	763	4.98%	9.96%	7.99%	11.14%
si	1544865 6	20.52%	24.63%	31.89%	32.84%
saxpy	9600000	0	0	0	0
sgemv	531000	0.38%	1.66%	2.41%	4.37%
mean		7.57%	9.44%	10.79%	11.86%

~~Having a predictive model makes a difference!~~



Publications

- A new search algorithm for optimization sequences in an adaptive compiler, Y. Guo, B. Zhang, K. Cooper and D. Subramanian, Rice University Tech. Report, 2006.
- ACME: Adaptive compilation made efficient, K. Cooper and A. Grosul and T. J. Harvey and S. Reeves and D. Subramanian and L. Torczon and T. Waterman, LACSI Symposium 2005.
- Exploring the structure of compilation sequences using randomized search algorithms, K. Cooper and A. Grosul and T. J. Harvey and S. Reeves and D. Subramanian and L. Torczon and T. Waterman, LACI Symposium, 2004.
- Compilation order matters, Technical Report, January 2002, Rice University (with K. Cooper, L. Torczon and T. Harvey).
- Adaptive optimizing compilers for the 21st century, *Journal of Supercomputing*, 2002 (to appear). Also appeared in the *Proceedings of the LACSI symposium*, Fall 2001. Here's the talk at the LACSI symposium, Fall 2001 (with K. Cooper and L. Torczon).
- Optimizing for reduced code space using genetic algorithms, *ACM SIGPLAN Workshop on Languages, Compilers, and Tools for Embedded Systems (LCTES)*, Atlanta, GA, 1999, will appear as an issue of SIGPLAN Notices) (with P. Schielke and K. D. Cooper).
- An experimental evaluation of list scheduling, Technical report, September 1998, Rice University (with K. Cooper and P. Schielke).



What have we learned?

- Amount of time spent on understanding underlying system.
- Amount of time spent on determining what to model.
- Prevalence of non-stationarity.
- Adapting models is expensive - need for lightweight mechanisms.

Scientific drivers

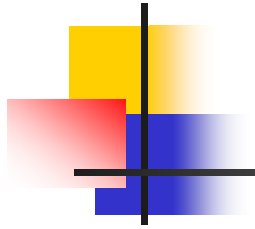


Science Magazine

The vision

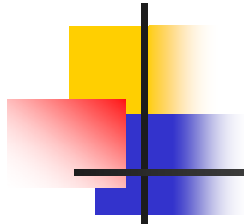
- "System identification" for large, non-stationary (distributed) systems.
- Off-the-shelf components for putting together feedback controllers with performance guarantees for such systems.





Collaborators

- Diana Gordon, ONR and Sandra Marshall, San Diego State University
- George Zouridakis, University of Houston
- Richard Stoll, Rice University
- Bradley Broom, Prahlad Ram, Tim McDonnell, MD Anderson Cancer Center
- Keith Cooper and Linda Torczon, Rice University



Students

■ Human learning

- Richard Thrapp, National Instruments
- Peggy Fidelman, PhD in CS/UT Austin
- Igor Karpov, PhD in CS/UT Austin
- Paul Ramirez
- Gwen Thomas, Green Hills
- Tony Berning
- Gunes Ercal (CRA mentee)
- Deborah Watt (CRA mentee)
- Scott Griffin, Rational
- Scott Ruthfield, Microsoft
- Chris Gouge, Microsoft
- Stephanie Weirich (Asst. Prof. at UPenn)
- Sameer Siruguri, MS, Purple Yogi
- Lisa Chang, MS, IBM
- Nuwan Rathnayake, Rice junior
- Ian Stevenson, Rice senior
- Farhan Baluch, University of Houston, MS 2006



Students

- Conflict
 - Michael Friedman
 - Adam Larson
 - Adam Stepinski, Rice sophomore
 - Clement Pang, Rice junior
 - Benedict Lee, MS 2007
 - Derek Singer, Rice junior
- Cancer
 - Jared Flatow, Rice junior
- Compilers
 - Phil Schielke, PhD 2002
 - Alex Grosul, PhD 2005
 - Todd Waterman, PhD 2005
 - Yi Guo, PhD 2007
 - Tim Harvey, post-doctoral fellow



Sponsors

- Conflict analysis: NSF ITR 0219673
- Adaptive compilers: NSF ITR 0205303
- Human learning: ONR N00014-96-1-0538
- Cancer: GC4R Foundation Grant