Advanced Learning Classifier Systems

Prof Will Browne

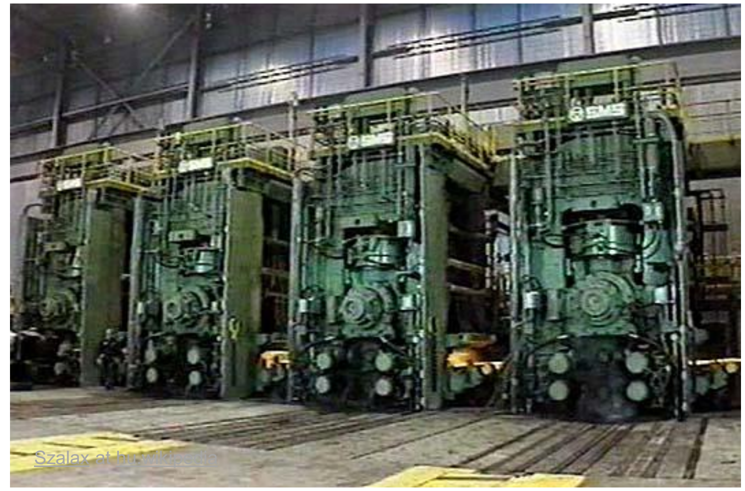
Will.Browne@qut.edu.au

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07…$15.00
https://doi.org/10.1145/3449726.3461429

1

Data-mining in a Steel Hot Strip Mill



2

Learning Classifier Systems

Abstracted Rules
e.g. 'if side guide setting < width, then poor quality product

Abstraction checks for patterns in the base rules and crates and abstracted rules for each discovered pattern

Base rules
e.g. if side-guide-setting = 80, width = 82 then poor quality product
if side-guide-setting = 79, width = 80 then poor quality product

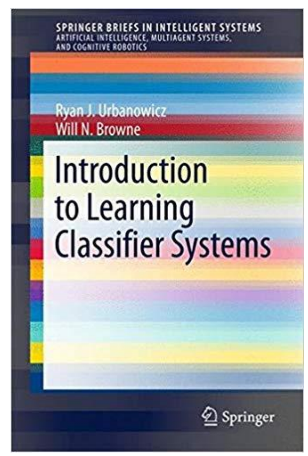
Learning system

Raw Data
e.g. Features

'side-guide setting'	' width'	:	'product quality'
78	81	:	poor
79	80	:	poor
78	76	:	good
…			

3

Learning Classifier Systems



- Ryan Urbanowicz faculty at the University of Pennsylvania in the Pearlman School of Medicine

- Will Browne is Professor at Queensland University of Technology.

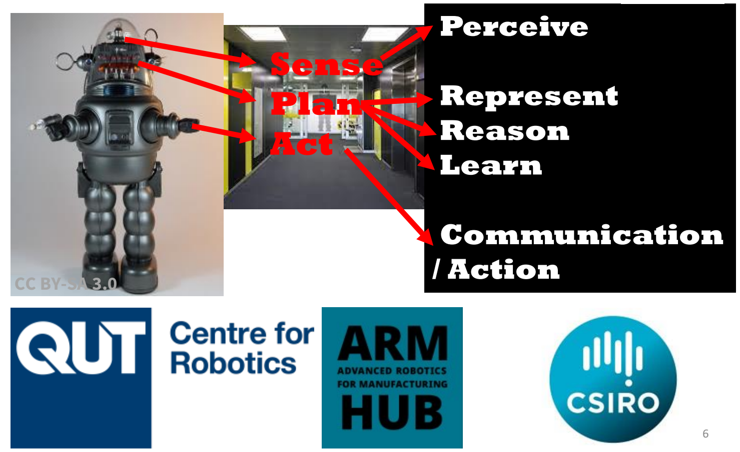
4

4

Research Team

5

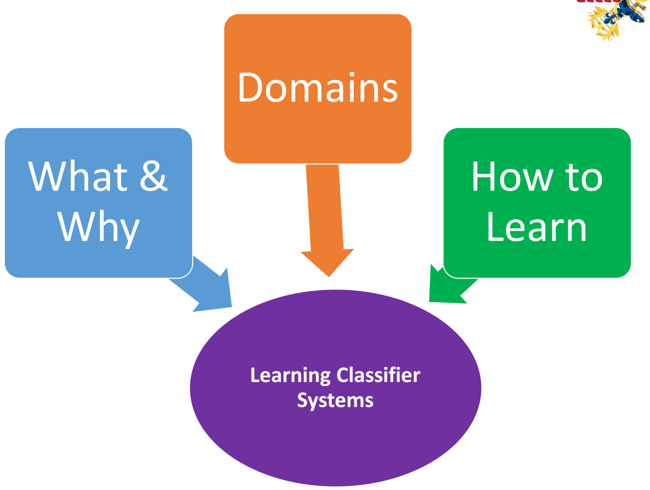
Learning Cognitive Systems?



6

6

Overview of Tutorial



7

7

Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

x Up-to-Date summary of all the excellent work in the field:
IWLCS — 24th International Workshop on Learning Classifier Systems
https://iwlcs.organic-computing.de

8

8

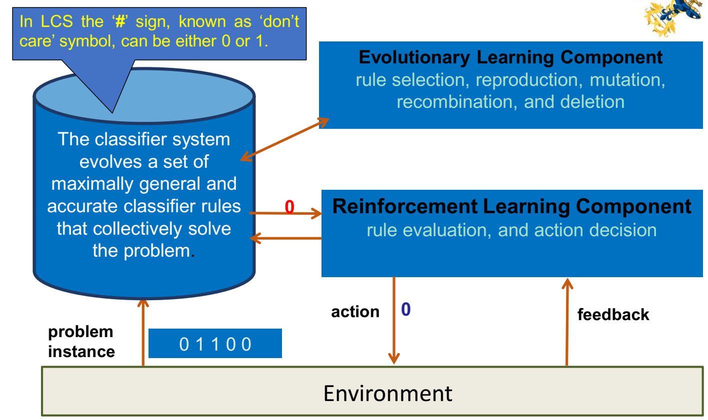
Take-Homes of Tutorial:

- What & Why
  - ***What are the important systems in the LCSs concept.***
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

9

9

LCS Overview



Isidro M. Alvarez - isidro.alvarez@ecs.vuw.ac.nz

10

10

Take-Homes of Tutorial:

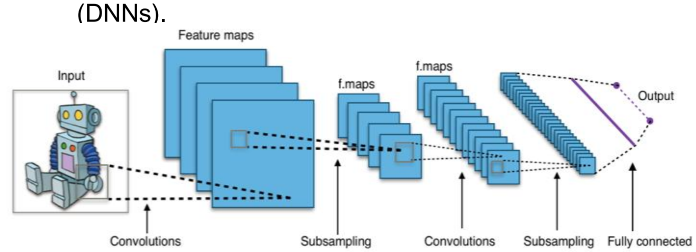
- What & Why
  - What are the important systems in the LCSs concept.
  - ***Why LCSs are important/useful,***
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

11

11

Connectionist learning

- While the very first AI systems were easily interpretable, the last years have witnessed the rise of **opaque** decision systems such as Deep Neural Networks (DNNs).



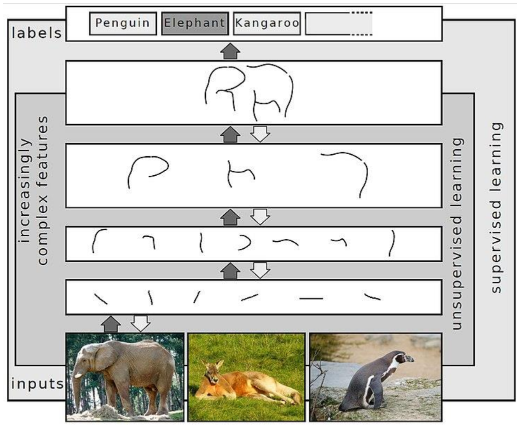
Aphex34 CC BY-SA 4.0

Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., ... & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82-115.

12

12

Opaque learning



13

13

Symbolic learning

Practical Application of a Learning Classifier System in a Steel Hot Strip Mill

- "NN might **learn** these rules, but not in **transparent** form.
- **Transparency** in the rule base is essential to allow operators, engineers and managers to validate and learn from the rules."

Browne, W., Holford, K., Moore, C., & Bullock, J. (1998). A practical application of a learning classifier system in a steel hot strip mill. In *Artificial Neural Nets and Genetic Algorithms* (pp. 611-614). Springer, Vienna.

14

14

LCSs are "Wondrous"

- ❖ Learning Classifier Systems combine the global search of *Evolutionary Algorithms* with the local optimisation of *Reinforcement Learning* to address classification and regression problems.
- ❖ The knowledge extracted though interacting with data or embedded in an environment is human readable.
- ❖ 'Inventing' as LCS' flexible nature allows application to many domains with many types of feedback on solution progress.

- ❖ Bit 'swampy' as an LCS is not a one line algorithm with independent methods and few parameters.

15

15

Take-Homes of Tutorial:

- **What & Why**
  - **What are the important systems in the LCSs concept.**
  - **Why LCSs are important/useful,**
- **Domains of application**
  - ***Requirements from different classification domains***
  - **eXplainable AI (XAI) using LCSs,**
- **How to Learn**
  - **Visualising learnt patterns**
  - **Combining blocks of knowledge**
  - **Constituent & holistic (lateralized) learning**
  - **Layered, continual and cognitive learning.**

16

16

135-Bit Multiplexer VS 10-Bit Majority-On

43,556,142,965,880,123,323,311,94
9,751,266,331,066,368

VS

1024

Which is harder to solve?

17

17

Symbol generation in LCSs

- Rules:

If <*conditions*> **then** <*action*>

Conditions relate to the features in the problem domain

Action relates to the target, e.g. class, movement of robot, …

If < feature 1 is true, feature 2 is false, feature 3 is true> then <Class A>

Rule- 101:A

If < f1 is true, f2 is true or false, f3 is false> then <Class B>

Rule- 1#0:B

18

18

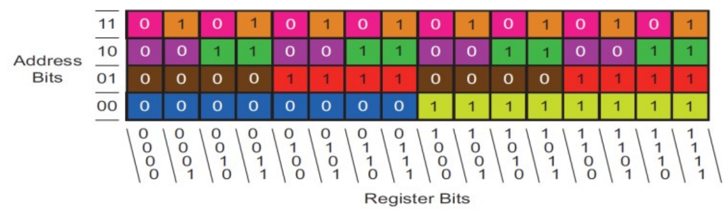
Representation: LCS spaces

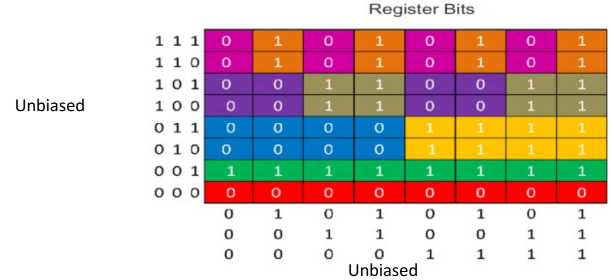
Example : Multiplexer

- Has six features and one action, binary coded [0, 1]
- Used in electronics for efficient input of data
- Samples include: 001000:1, 000111:1, 111111:1
- Given just the data samples can the relationship between Conditions and Acton be learned?

19

Representation: LCS search spaces

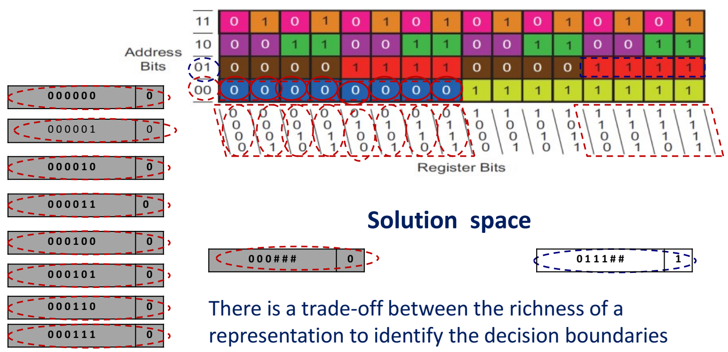




20

Representation: LCS spaces

Sample space



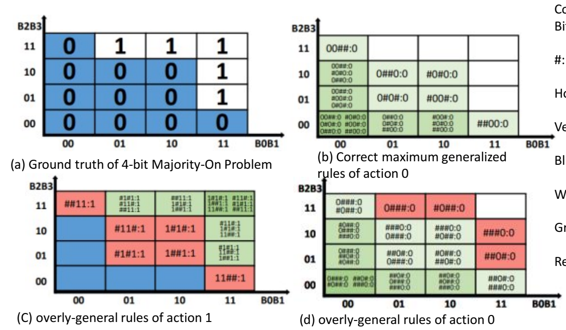
There is a trade-off between the richness of a representation to identify the decision boundaries in a search space and the size of the solution space.

absiddique@ecs.vus.ac.nz

Overly Specific Issue

- Large number of overly specific plugs a population when evaluation.
- Basic Accuracy-based LCSs produced overly specific rules
- Addressed by the Subsumption method.

Overly General Issue



Condition form: Bit0Bit1Bit2Bit3

#: don't care, cover 1 or 0

Horizontal axis: Bit0 Bit1

Vertical axis: Bit2 Bit3

Blue: uncovered action 0

White: uncovered action 1

Green: Correct covered

Red: Incorrect covered

Optimal solution(s)

Describes an ideal ruleset rather than the requirement of individual member rules; multiple [*O*]s exist;

Previous: [*O*] set

10##1#:1 10##0#:0
11###1:1 11###0:0
01#1##:1 01#0##:0
001###:1 000###:0

Completeness;
Correctness;
Minimality;
Non-overlapping;

Proposed: Natural Solution

Describes the member rules rather than the ruleset;
A dataset only has one natural solution;

Consistent;
Un-subsumable;
Allow overlapping;

Choice of Encoding for real numbers

Euclidean and Hamming distances alter search space

Integer	Binary	Gray	Enumerated
0	000	000	0000000
1 :1	001:1	001 :1	0000001 :1
2 :1	010:2	011 :1	0000011 :1
3 :1	011:1	010 :1	0000111 :1
4 :1	100:3	110 :1	0001111 :1
5 :1	101:1	111 :1	0011111 :1
6 :1	110:2	101 :1	0111111 :1
7 :1	111:1	100 :1	1111111 :1

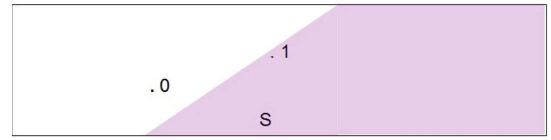
Encoding :Hamming distance

How to encode the range: 0 → 3 and 0 → 4 ?

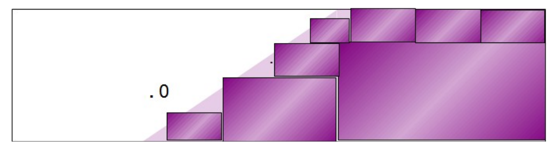
27

27

Oblique Class boundaries



We have a search space with only two classes to identify: 0 & 1
It's real numbered so we decide to use bounds: e.g. $0 \le x \le 10$



We form Hypercubes / Hyperrectangles, but these are not often suited to oblique domains
Imagine sine wave domains.....

28

Alternative representations

Many other representations available

- Artificial neural networks
- Fuzzy logic/sets
- Horn clauses and logic
- S-expressions, GP-like trees and code fragments.
  - Is a LCS with S-expressions not just GP? NO!
  - How to tailor functions without introducing bias?
  - How to identify building blocks of Subexpressions?
  - When are two Subexpressions equivalent?
- Is trade-off between reduced problem search space to increased solution search space worth it?

29

Symbolic Regression

Name	Problem Equation
Nguyen-01	$x^3 + x^2 + x$
Nguyen-03	$x^5 + x^4 + x^3 + x^2 + x$
Nguyen-04	$x^6 + x^5 + x^4 + x^3 + x^2 + x$
Nguyen-05	$sin(x^2) * cos(x) - 1$
Nguyen-06	$sin(x) + sin(x + x^2)$
Nguyen-07	$ln(x+1) + ln(x^2+1)$
Nguyen-08	$sqrt(x)$
Nguyen-09	$sin(x) + sin(y^2)$
Nguyen-10	$2 * sin(x)cos(y)$

Task: Symbolic Regression

- Success if < 0.01 error
- Solved if 'exact' function learnt

Nguyen-10
(sin(sin(sin((sin(x1 + sin(x1 + x1)) + sin(x1 + (x1 + x1))) + x2) + (sin(sin(x1 + x1)) + sin(sin(sin(x1 + x1) + sin(sin(x1 + x1) + (x1 + x1)))))) + (x1 + sin(sin(sin(sin(sin(x1))))))) + sin((sin(x1 + sin(sin(sin(sin(sin(x1 + x1)) + sin(x1 + x2))))) + sin(sin(sin(sin(x1 + x1) + x2)) + sin((sin(x1 + sin(x1 + x2)) + x1) + sin((x1 + x1) + x1)))) + x2)) * x1

Nguyen-10 -1.00, 0.30 -0.61, 0.61 : →
D0 1.0 / sin D0 1.0 / sin + D0 0.0 * D1 D0 * - D0 1.0 / D1 D0 * / / +

30

30

Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - *eXplainable AI (XAI) using LCSs,*
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

31

31

DARPA - XAI

- Dramatic success in machine learning has led to a torrent of Artificial Intelligence (AI) applications.
- Continued advances promise to produce autonomous systems that will perceive, learn, decide, and act on their own.
- However, the effectiveness of these systems is limited by the machine's current inability to explain their decisions and actions to human users

https://www.darpa.mil/program/explainable-artificial-intelligence

32

32

eXplainable AI (XAI)?

- The results of the solution can be understood by humans

Or

- **Understandability** – humans can know how the model works
- **Comprehensibility** - represent its learned knowledge in a human understandable fashion
- **Interpretability** - to provide meaning
- **Explainability** - interface between humans and AI
- **Transparency** - if by itself it is understandable.

33

33

Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - *Visualising learnt patterns*
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

34

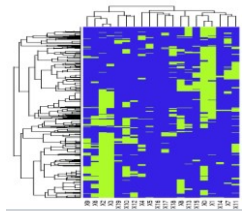
34

Pattern visualisation

Clustering are based on Agglomerative Hierarchical Clustering (AHC)

Clustering are based on rules' generalization level

Previou... ...gue patterns



Prop... ...erns

An [O] for 6-bits Multiplexer Problem:

Bit0	0	0	1	1	0	0	1	1
Bit1	0	1	0	1	0	1	0	1
Bit2	1	#	#	#	0	#	#	#
Bit3	#	1	#	#	#	0	#	#
Bit4	#	#	1	#	#	#	0	#
Bit5	#	#	#	1	#	#	#	0
Action	1	1	1	1	0	0	0	0

Number of rules in Clusters (C)

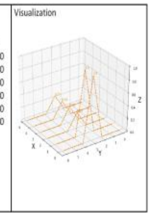
C0	C1	C2	C3	C4	C5	C6
0	0	0	8	0	0	0

Statistics:

Cluster ID (Y)

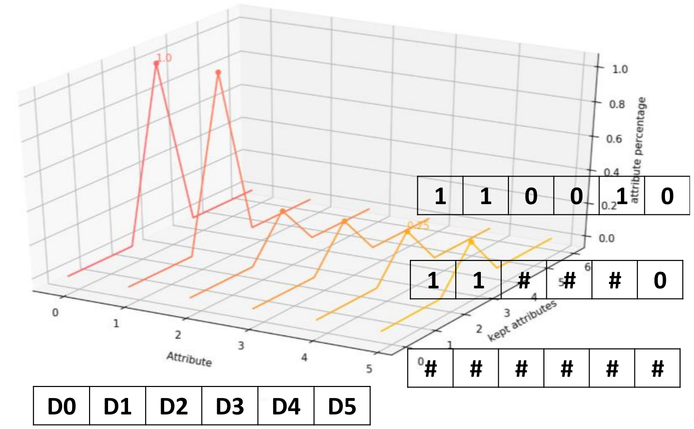
Bit ID (X)	0	1	2	3	4	5	6
0	0.00	0.00	0.00	1.00	0.00	0.00	0.00
1	0.00	0.00	0.00	1.00	0.00	0.00	0.00
2	0.00	0.00	0.00	0.25	0.00	0.00	0.00
3	0.00	0.00	0.00	0.25	0.00	0.00	0.00
4	0.00	0.00	0.00	0.25	0.00	0.00	0.00
5	0.00	0.00	0.00	0.25	0.00	0.00	0.00

Visualization



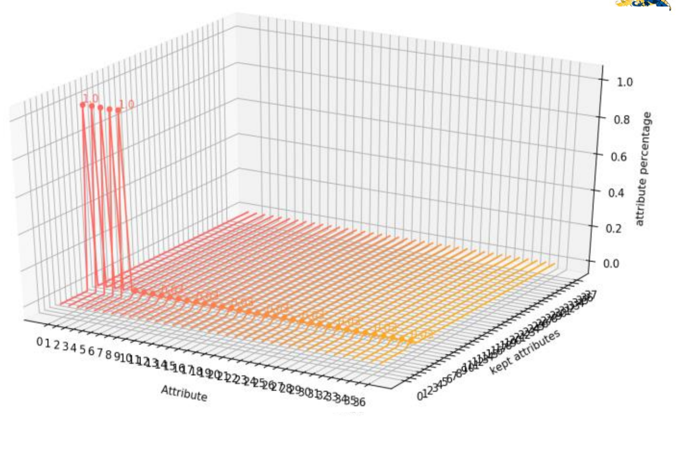
35

Identification of Problem Patterns



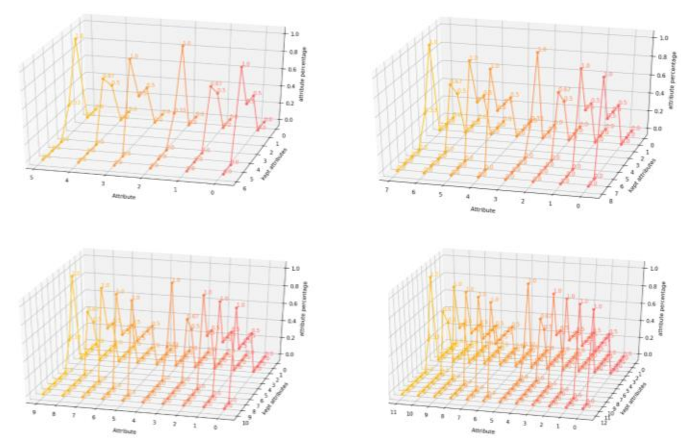
36

Visualized VS Standard XCS Visualized



37

Purpose of Problem Patterns



38

Discovered size of optimal set in addressed n-bit domains

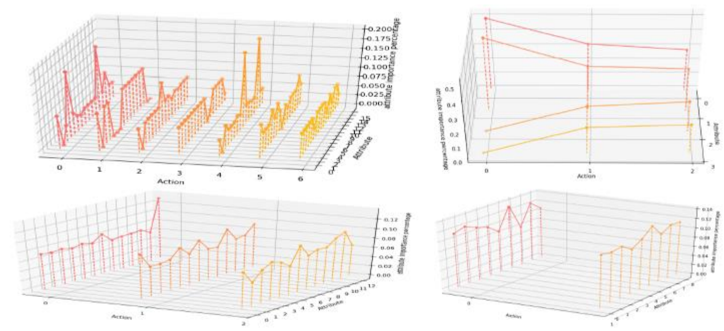
- Multiplexer: 2^m — m: the number of address bits
- Carry: $2^{\frac{N}{2}+1} + \sum_{K=1}^{K=\frac{N}{2}-2} * 2^K$ — N: the number of involved bits (N>4)
- Majority-On (Even): $C_N^{\frac{N}{2}+1} + C_N^{\frac{N}{2}}$
- Majority-On (Odd): $2 * C_N^{\frac{N+1}{2}}$

Majority-On is two different problems depending on N being even or odd!

39

39

RCR Visualized Results (Real Domains)

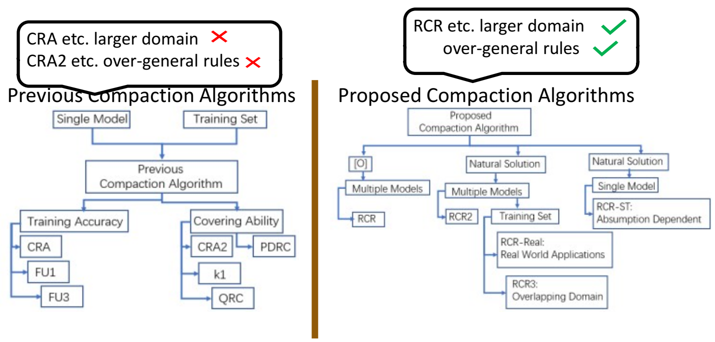


From left to right first line: ZOO, IRIS, second line: Wine, WBCD

40

40

Identify the Optimal Solution



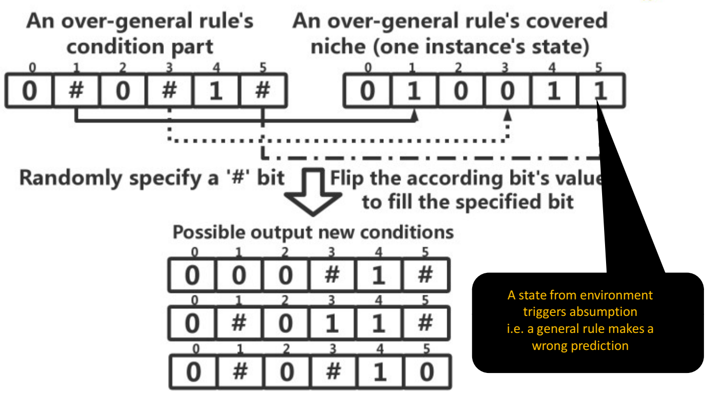
41

Compaction Testing Accuracy

Domain	CRA	FU1	FU3	CRA2	K1	QRC	PDRC	RCR	RCR2	RCR3
Zoo	0.58	0.52	0.95	0.92	0.19	0.94	0.92	0.98	0.98	0.93
Mushroom	0.99	0.99	0.99	0.99	0.34	1.0	0.99	1.0	1.0	0.99
German	0.63	0.65	0.63	0.63	0.57	0.67	0.63	0.71	0.71	0.64

42

Absumption: Specification



Improve LCS: ASCS (Training Accuracy)

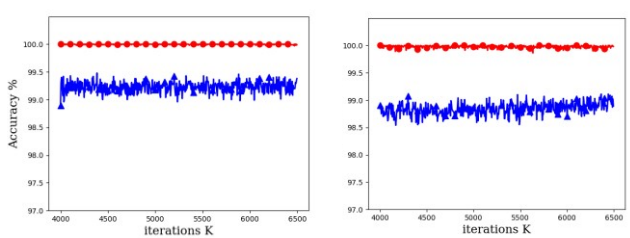
Domain	UCS (rules)	UCS (optimal)	ASCS (rules)	ASCS (optimal)
20 MUX (32)	6786	99%	108	100%
37 MUX (64)	7174	71%	320	100%
70 MUX (128)	13488	30%	[illegible]	[illegible]
10 Carry (78)	4954	56.2	[illegible]	[illegible]
12 carry (158)	6978	26.3	[illegible]	[illegible]
14 carry (318)	8968	14.9	[illegible]	[illegible]
12 Majority-On (1716)	8490	17%	[illegible]	100%
13 Majority-On (3432)	8739	7%	64	100%
14 Majority-On (6435)	10747	3.6%	12870	100%

ASCS can directly produce complex model to represent an explored domain, e.g. using 12870 different interactive rules to construct an optimal solution

Training Performance

12 Carry:

10 Majority-On:



Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - *Combining blocks of knowledge*
  - Constituent & holistic (lateralized) learning
  - Layered, continual and cognitive learning.

Symbol generation & its transfer

- Modeling a capacity to generate and then reuse symbols and functionalities
  - Different problems in the same domain are likely to contain common patterns
  - Patterns in one domain are useful in related domains
- XCSCFC [Iqbal, Browne 2014]

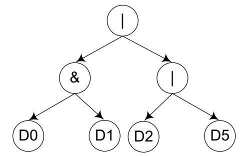

  - Generate GP-like logics as a hierarchical combination of symbols from basic symbols
  - Based on the XCS's powerful search capacity

47

47

Code Fragments

- Since 2011 - Code Fragments
- On-line, reinforcement learning
- Reusable + Rich Alphabet
- Part solutions and solution parts
- Condition bits - Independent number, location
- Action bits – State machines, computed actions



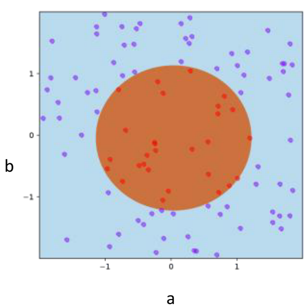
CF Leaf nodes: Features from the environment or other CF

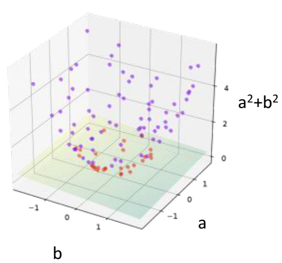
CF Root nodes: Originally pre-defined functions

Muhammad Iqbal 49

49

Representation: LCS search spaces





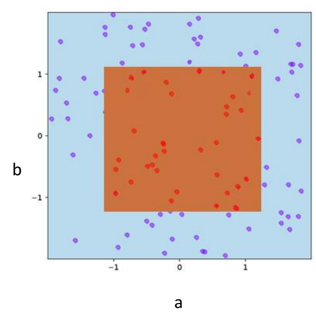
Classifier: $a^2+b^2 < 1$, for class 'red'

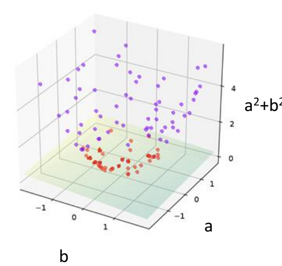
Code: ((a*a)+(b*b)) < 1

Code Fragment Rules: aa*bb*+1< : red
aa*bb*+1>= : blue

50

Representation: LCS search spaces





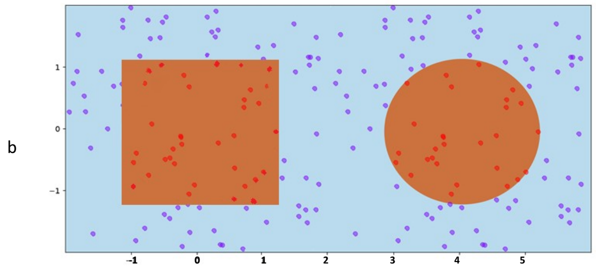
Classifier: $a^2 < 1$ AND $b^2 < 1$,

Code: ((a*a)<1)&&((b*b) < 1)

Code Fragment Rules: aa*1<bb*1<&&1< : red
aa*1<bb*1<&&1 >= : blue

51

Representation: LCS search spaces



Classifier: if a < 2, $a^2 < 1$ AND $b^2 < 1$, then red
if a >= 2, $(a-4)^2+b^2 < 1$, then red

Code: if a<2 AND ((a*a)<1)&&((b*b) < 1), else if ((a*a)+(b*b)) < 1

Code Fragment Rules:
a 2 <, a4-a*bb*+1< : red
a 2 <, a4-a*bb*+1>= : blue
a 2 >=, aa*1<bb*1<&& 1< : red
a 2 >=, aa*1<bb*1<&& 1>= : blue

52

Code fragments

Rule-condition is a set of GP-like trees

Condition						Action
D0D0~|	D0D5d~	D1D4r~	D0D0~|	D0D0~|	D0D0~|	0

Where features (D0, … , D5) are the leafs of the trees, functions at nodes

53

Building blocks for knowledge

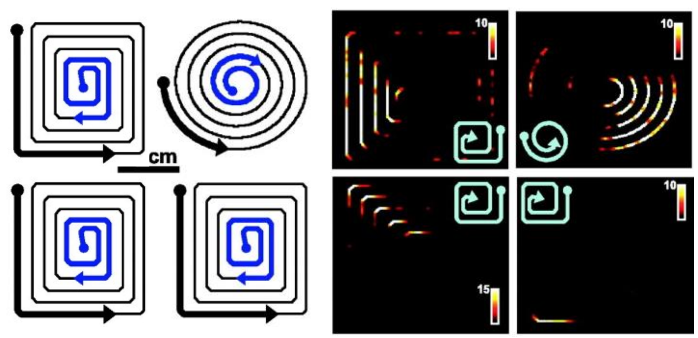
- Improvement of performance through experience
- Knowledge gained through experience
- EC is very good at searching, e.g. for building blocks but …

Isidro M. Alvarez -
isidro.alvarez@ecs.vuw.ac.nz

54

54

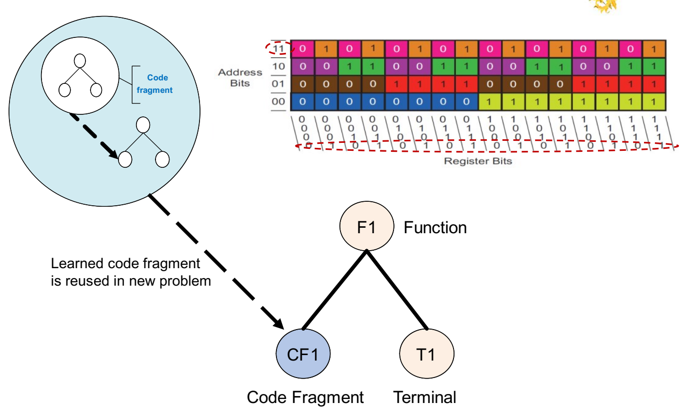
Building block evidence – Doug Nitz



http://www.dnitz.com/#research
used with permission

55

Code Fragment Reuse



Reusing the Extracted Knowledge

Problem Level	Code Fragments	
	Name	Expression
Level 1	L1_0	D1 D0 D4 d r
	L1_1	D5 ~ D1 D0 & &
	...	...
Level 2	L2_0	L1_15 D2 L1_4 r &
	L2_1	L1_5 D2 | L1_11 D3 & r
	...	...
Level 3	L3_0	L2_9 L1_7 D2 | r
	L3_1	L1_10 L2_17 | L2_1 L2_1 r &
	...	...

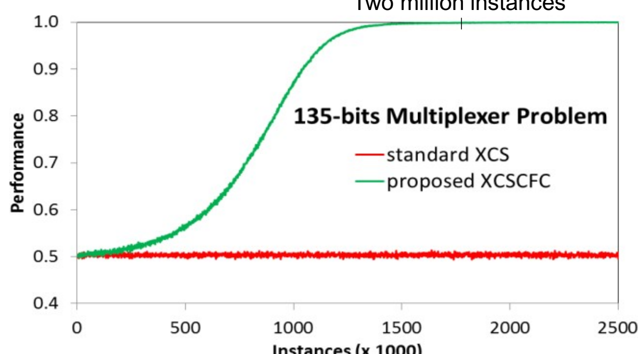
Muhammad.Iqbal@ecs.vuw.ac.nz

XCS with Code-Fragment Conditions

Total possible instances = 4 × 10^{40}

Sampling only 1 in 10^{34} instances.

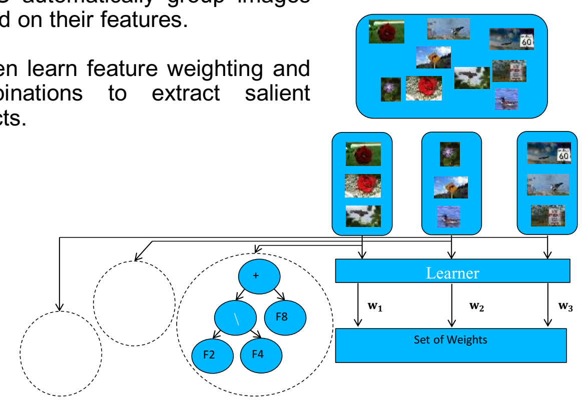
Two million instances



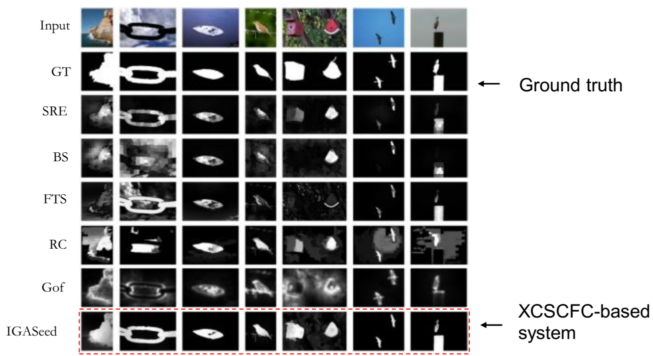
Muhammad.Iqbal@ecs.vuw.ac.nz

Salient object detection with XCSCFC

- LCS automatically group images based on their features.
- Then learn feature weighting and combinations to extract salient objects.



Salient object detection with XCSCFC



60

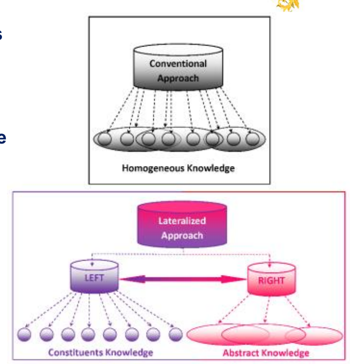
Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - *Constituent & holistic (lateralized) learning*
  - Layered, continual and cognitive learning.

61

Lateralized AI Approach

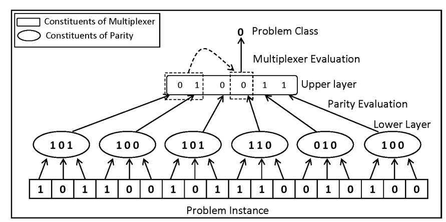
- **Left half** considers individual features and simple niches
- **Right half** creates abstract knowledge representation, i.e. high order features extracted across niches
- **Input Signal Processing**
  - A new input signal is placed in the context of system knowledge
  - Attention is given to the more salient parts of a signal

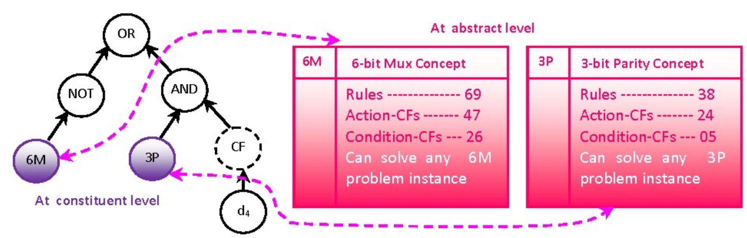


absiddique@ecs.vuw.ac.nz

62

Interpretation of Results





63

Interpretation of Results

Is learning the *n*-th thing any easier than learning the first?
Life-long learning, Thrun 1996

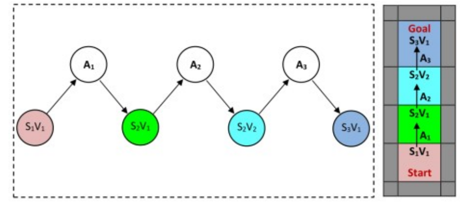
D_0	D_1	D_2	D_3	D_4	D_5	D_6	P6	P7	Rule1 if ~(~D_6) -> P7=~P6		Rule2 If ~D_6 -> P7 = P6	
									Applicable	Output	Applicable	Output
0	0	0	0	0	0	0	0	0	×		✓	0
0	0	0	0								×	-
0	0	0	0								✓	1
0	0	0	0	0	1	1	1			0	×	-
0	0	0	0	1	0	0	1	1	×	-	✓	1
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
1	1	1	1	1	1	0	0	0	×	-	✓	0
1	1	1	1	1	1	1	0	1	✓	1	×	-

R-1: If NOT(NOT D_6) Then NOT P6
R-2: If NOT D_6 Then P6

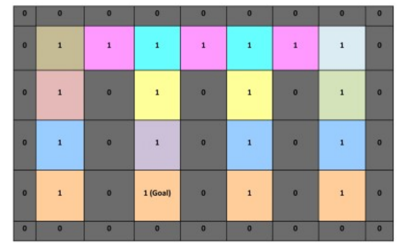
Almost, 7-bit parity is as easy as 2-bit parity if know 6-bit parity!

absiddique@ecs.vuw.ac.nz

Aliased Navigation Domain

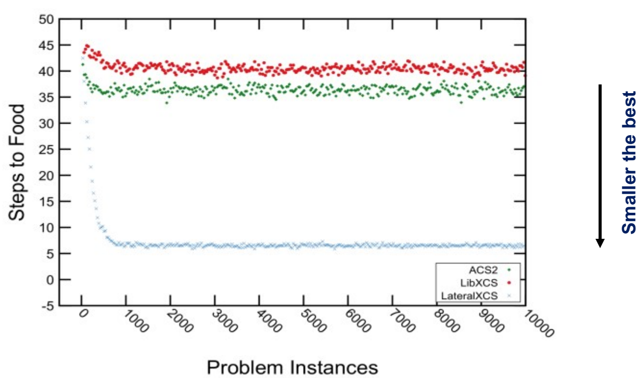


- Maze 10



Navigation Experiments

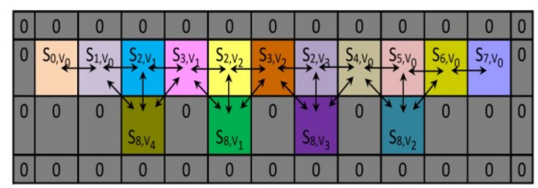
- Maze 10



Navigation Experiments

- L57

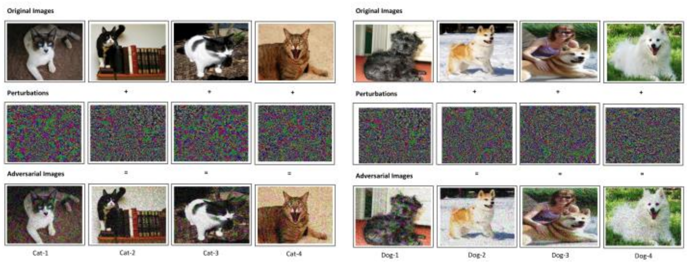
0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	1	0	1	0	1	0	F	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0



S_0=00100000 S_1=00110010 S_2=00101010 S_3=00110110 S_4=001F0110
S_5=0010F010 S_6=00100F10 S_7=00100010 S_8=11000001

Computer Vision Domain

➢ **Separating Cats from Dogs, even after Adversarial Attacks**



https://www.kaggle.com/c/dogs-vs-cats

68

Interpretation of Results

VGG Dog

Cat-1

Lateralized System Cat

Image Name	Whole Image Prediction (DL Model)	Constituents Predictions (DL-Models)	Constituent Predictions (LCSs-Models)
Cat-1	33.05 % Cat	99% CM, 99% DM	87.61% CM, 12.39% DM

69

Take-Homes of Tutorial:

- What & Why
  - What are the important systems in the LCSs concept.
  - Why LCSs are important/useful,
- Domains of application
  - Requirements from different classification domains
  - eXplainable AI (XAI) using LCSs,
- How to Learn
  - Visualising learnt patterns
  - Combining blocks of knowledge
  - Constituent & holistic (lateralized) learning
  - *Layered, continual and cognitive learning.*

70

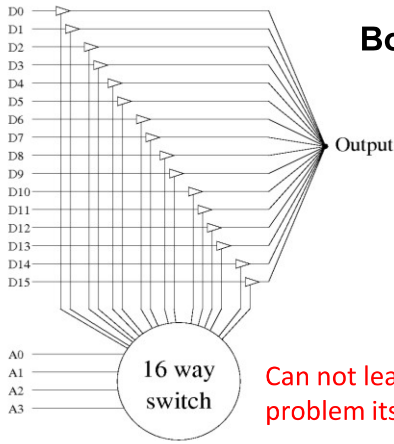
Layered Learning

- One unwieldy problem split into several steps
- Each step feeds into the next
- Educator – Instructs using **Threshold Concepts**

Isidro M. Alvarez - isidro.alvarez@ecs.vuw.ac.nz

71

Boolean Multiplexer

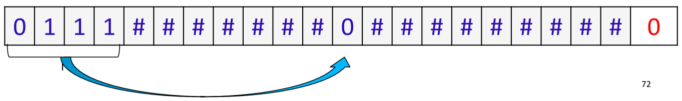


Length $n = k + 2^k$

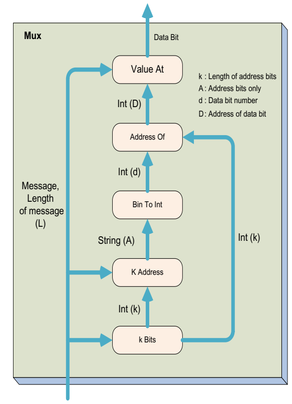
Humans need to know:

- Recognise Patterns
- Localisation
- Binary numbers (powers and sums)
- Addressing
- Returning a value

Can not learn these within the problem itself.



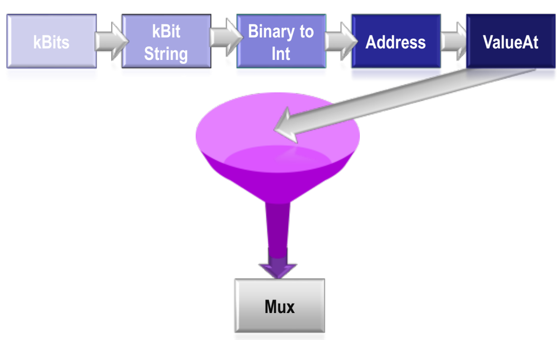
Layered-learning in LCSs



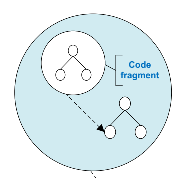
- Train system in 5 stages
- KBits, KBitString, BinToInt, Address, ValueAt
- Mux: address bits, data bits
- Length of a Mux given by:
  - $L = k + 2^K$
- K address bits given by:
  - K = $\lfloor Log_2 L \rfloor$

Training Path: Layered Learning

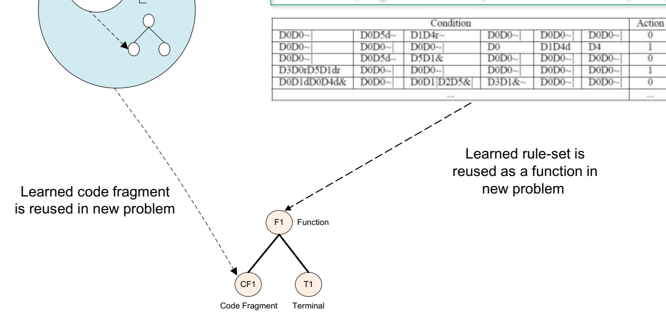
- Sub-problem solutions feed Mux problem



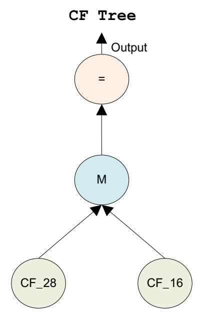
Rule-sets as Functions



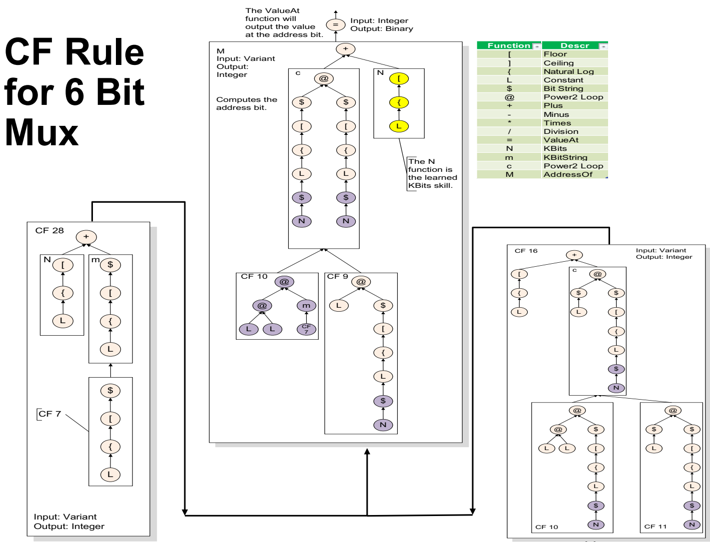
Concept:
If <Conditions> Then <Actions>
If <Input> Then <Output>
Function(Arguments <Input> Return <Output>)



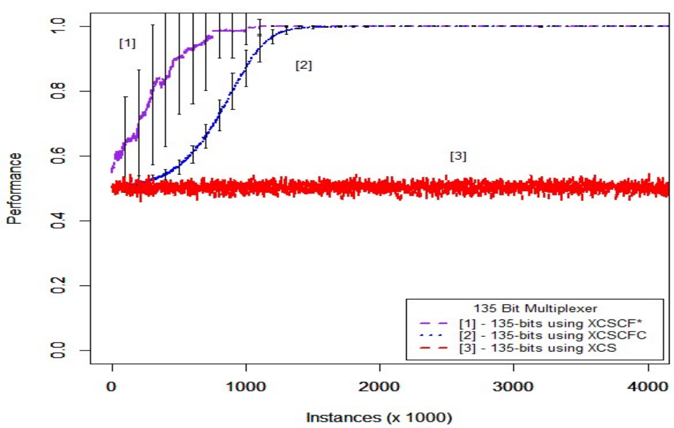
CF Rule for 6 Bit Mux



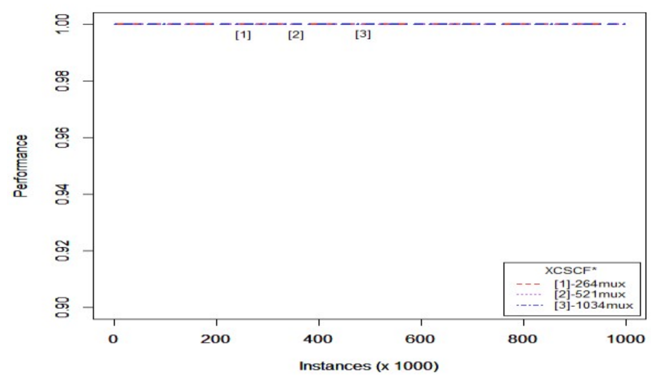
CF Rule for 6 Bit Mux



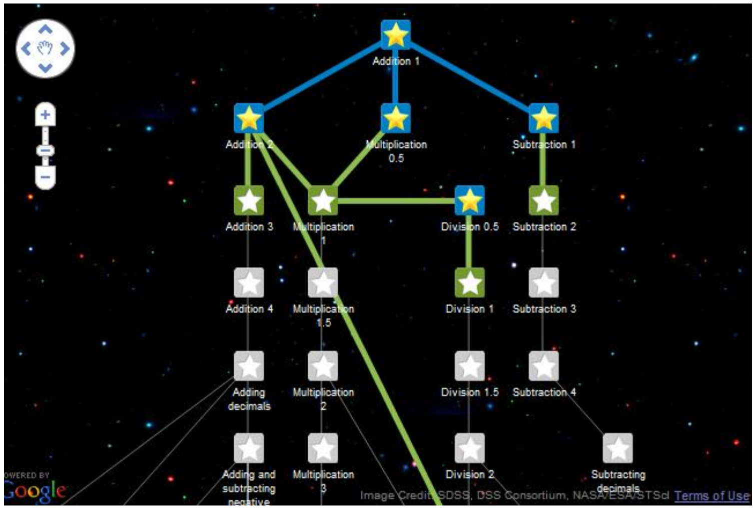
Training 135 Bit Mux



Testing 6 Bit Rules on 264, 521, 1034 Bit Mux



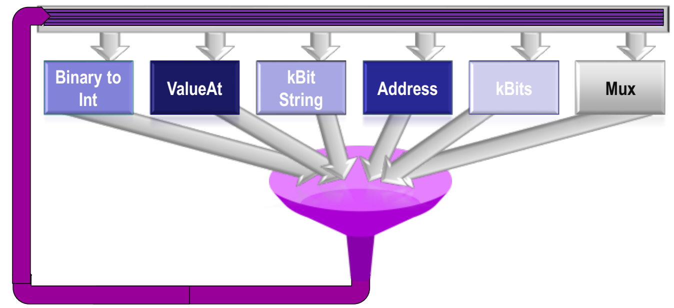
Continuous Learning



80

Transfer Learning

- All problem solutions could feed unsolved problems

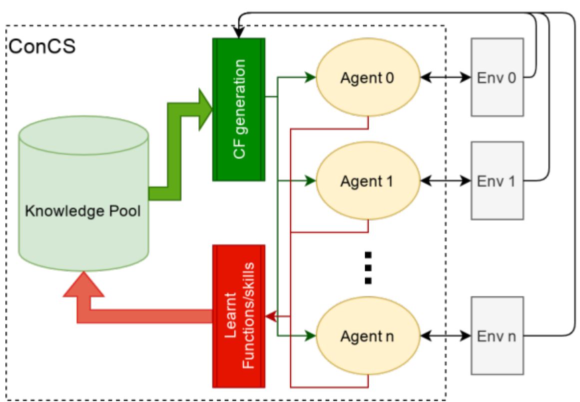


Isidro M. Alvarez - isidro.alvarez@ecs.vuw.ac.nz

81

81

Continual Learning Classifier System: ConCS



Agent: type-fitting XCSCFA

82

82

Solve problems concomitantly

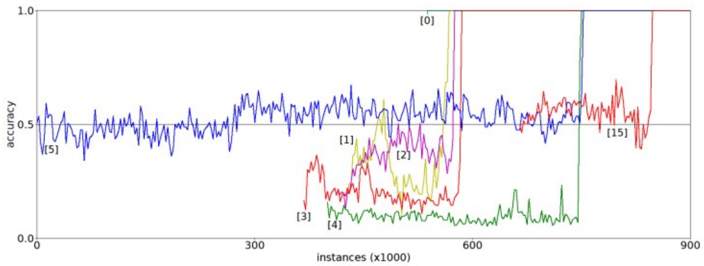
Id	Functions (abbreviations)	Inputs	Input types	Output type	Anticipated operation to be learnt
0	Address Length given Mux size	x_0	integer	integer	$\lfloor log_2(x_0) \rfloor$
1	Address Length given Mux bitstring	x_0	list	integer	$\lfloor log_2(len(x_0)) \rfloor$
2	Address Bits	x_0	list	list	$x_0[0 : \lfloor log_2(len(x_0)) \rfloor]$
3	Decimal value of Address Bits	x_0	list	integer	$bin2dec(x_0[0 : \lfloor log_2(len(x_0)) \rfloor])$
4	Data Bit Position	x_0	list	integer	$\lfloor log_2(len(x_0)) \rfloor +$ $bin2dec(x_0[0 : \lfloor log_2(len(x_0)) \rfloor])$
5	General Multiplexer (mux)	x_0	list	Boolean	$x_0[\lfloor log_2(len(x_0)) \rfloor +$ $bin2dec(x_0[0 : \lfloor log_2(len(x_0)) \rfloor])]$
6	Half String Size	x_0	list	integer	$len(x_0)/2$
7	First Half	x_0	list	list	$x_0[0 : (len(x_0)/2)]$
8	Second Half	x_0	list	list	$x_0[(len(x_0)/2) : len(x_0)]$
9	Binary Addition of 2 halves	x_0	list	list	$x_0[0 : (len(x_0)/2)] \oplus$ $x_0[(len(x_0)/2) : l]$
10	Length of Binary Addition	x_0	list	integer	$len(x_0[0 : (len(x_0)/2)]$ $\oplus x_0[(len(x_0)/2) : l])$
11	General Carry-one ($carr$)	x_0	list	Boolean	$len(x_0[0 : (len(x_0)/2)] \oplus$ $x_0[(len(x_0)/2) : l]) > len(x_0)/2$
12	Sum Modulo 2	x_0	list	Boolean	$sum(x_0)\%2$
13	General Even-parity ($epar$)	x_0	list	Boolean	$sum(x_0)\%2 = 0?$
14	General Majority-on (maj)	x_0	list	Boolean	$sum(x_0) > len(x_0)/2?$
15	Hierarchical Multiplexer	x_0	list	Boolean	$mux(loop(epar, x_0, 3))$

83

83

Solve problems concomitantly

ConCS: learning curve on Hierarchical Multiplexer domain and its subproblems with random arrivals



0-5: subproblems of Multiplexer; 15 Hierarchical Multiplexer

84

84

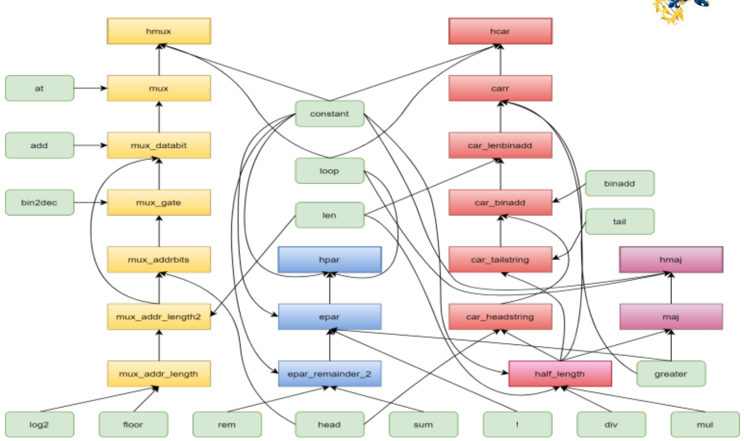
Solve problems concomitantly

Functions & Skills	Function Name	Learned Solutions
Address Length given Multiplexer size	mux_addr_length	$floor(log2(x_0))$
Address Length given MUX attributes	$mux\_addr\_length2$	$mux\_addr\_length(len(attlst))$
Address Bits	$mux\_addrbits$	$head(attlst, mux\_addr\_length2(attlst))$
Decimal value of Address Bits	mux_gate	$bin2dec(mux\_addrbits(attlst))$
Data Bit Position	$mux\_databit$	$add(mux\_gate(attlst),$ $mux\_addr\_length2(attlst))$
Variable-size Multiplexer	mux	$@(attlst, mux\_databit(attlst))$
Hierarchical Multiplexer	$hpar$	$mux(loop(epar, x_0, 3))$
Sum Modulo 2	$epar\_mod\_2$	$mod(sum(attlst), 2)$
Variable-size Even-parity	$epar$	$\neg(epar\_mod\_2(attlst))$ $greater(c(1), epar\_mod\_2(attlst))$ $greater(div(i,k), epar\_mod\_2(attlst)$ $(i \leq k)$
Hierarchical Even-parity	$hpar$	$epar(loop(epar, x_0, 3))$ $epar(loop(epar, x_0, 1))$
Half String Size	$half\_length$	$div(len(attlst), 2)$ $mul(len(attlst), div(c(i), c(2i)))$
First Half	$car\_headstring$	$head(attlst, half\_length(attlst))$
Second Half	$car\_tailstring$	$tail(attlst, half\_length(attlst))$
Binary Addition of two halves	car_binadd	$binadd(car\_headstring(attlst),$ $car\_tailstring(attlst)))$
Length of Binary Sum	$car\_lenbinadd$	$len(car\_binadd(attlst))$
Variable-size Carry-one	$carr$	$greater(car\_lenbinadd(attlst),$ $half\_length(attlst))$
Hierarchical Carry-one	$hcar$	$carr(loop(epar, x_0, 3))$
Variable-size Majority-on	maj	$greater(sum(x_0), half\_length(attlst))$
Hierarchical Majority-on	$hmaj$	$maj(loop(epar, x_0, 3))$

85

85

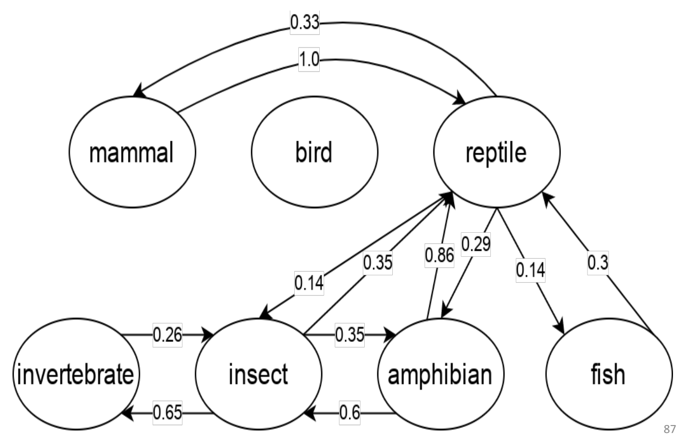
ConCS: knowledge network



86

86

Relatedness on UCI Zoo



87

87

Summary of Advanced LCSs

- **eXplainable AI**
  - Readable rules form a model, learned knowledge in CFs, can explain decisions, with time it is understandable.
- **Exact, visualisable patterns**
  - Interrogate meaning and scaling within large sample/solution space problems with interacting rules.
- **Lateralized learning**
  - Consider constituent and holistic knowledge at different levels of abstraction simultaneously.
- **Continual learning classifier systems**
  - Multitask learning while learning a curricula.

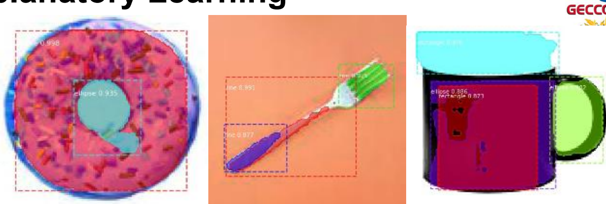
88

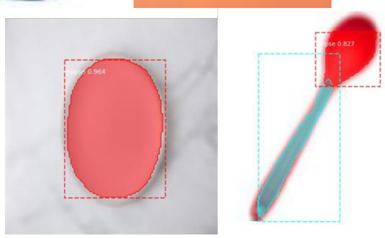
88

Explanatory Learning

89

Explanatory Learning





IF shape2 == null, ellipse or rectangle AND magnitude 2 is > 40.4 THEN spoon

IF shape1 == ellipse or rectangle THEN plate

IF shape2 == ellipse THEN plate

90

Getting there?

"the results of computer induction should be symbolic descriptions of given entities, semantically and structurally similar to those a human expert might produce observing the same entities.

Components of these descriptions should be comprehensible as single 'chunks' of information, directly interpretable in natural language, and should relate quantitative and qualitative concepts in an integrated fashion"

— R. S. Michalski, A theory and methodology of inductive learning, in Machine learning, Springer, 1983, pp. 83–134

91

91

Cognitive System

LCSs have a role as Cognitive Systems:

- Perceive — - Applicable to a wide range problems
- Represent, Reason, Learn — - Exceptionally flexible framework
- Communication / Action — - Transparent and reusable solutions

What's missing?
Memory to only consider relevant details of problems
Epochs to allow for parallel cloud computing