Learning Classifiers

Lecture 8

MONASH UNIVERSITY CLAYTON'S SCHOOL OF INFORMATION TECHNOLOGY

Summary

Reading: Eiben& Smith Chapter 7

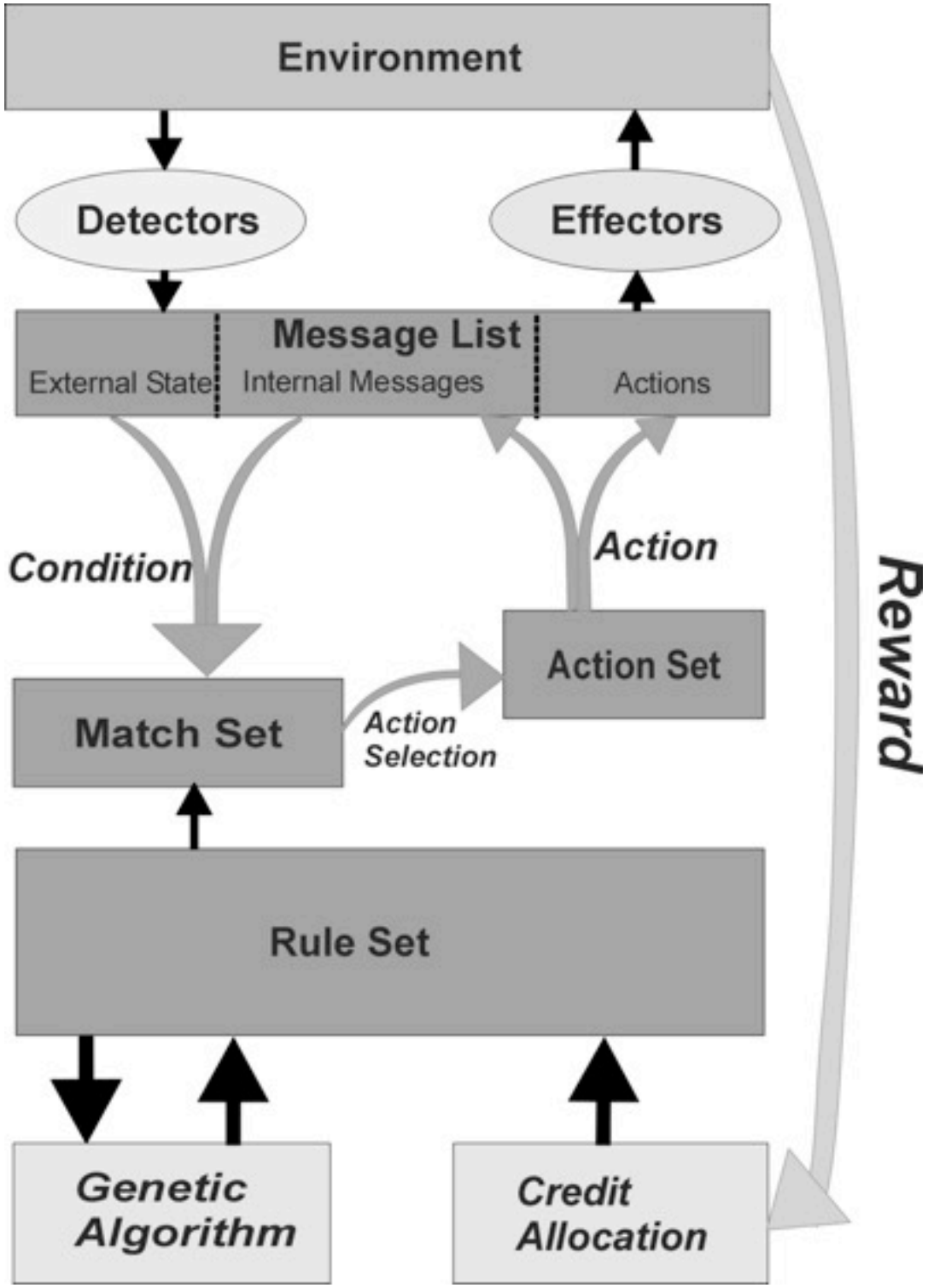
- Reinforcement learning
- Basic LCS
- ZCS – a "Zeroth-Level" classifier system
- XCS classifier algorithm
- Extensions
- Anticipatory LCS

Classifier Systems: Overview

- Evolutionary Machine Learning Method, originally developed by John Holland
- Use Rule Sets to represent knowledge
- Use an evolutionary algorithm for rule discovery, trying to evolve a system that maximizes future reward from the environment.
- LCS algorithms often demonstrate cooperation between population members (rule sets).
- "Pittsburgh style" has populations of rules sets, which are recombined. "Michigan style" has a single rule set, *fitness-sharing* (ZCS) and *accuracy-based* (XCS)

Overview: generic LCS

Representation	condition:action:prediction-tuples, conditions use {0,1,#} alphabet
Recombination	One-point crossover on conditions/actions
Mutation	Binary/ternary resetting as appropriate on action/conditions
Parent Selection	Fitness Proportional with sharing within environmental niches
Survivor Selection	Stochastic, inversely related to number of rules covering same environmental niche


Environment
Detectors
Effectors
Message List
External State
Internal Messages
Actions
Condition
Action
Reward
Action Set
Match Set
Action
Selection
Rule Set
Genetic
Algorithm
Credit
Allocation

LCS Details

- Environment state is transmitted to a set of **detectors** whose output is put on a **message list**.
- Message list may contain outer messages posted as a result of rules that have fired in a previous cycle or detector signals from previous cycles. This represents a form of memory.
- The condition part of each rule is examined for matching in the message list. Rules that match form a **match set** for this cycle.
- For each rule in the match set, rules **bid** based on their **specificity** and **strength.**
- The highest bidder adds its action to the **action set**, all of the rules that advocated that action are tagged as belonging to the action set for that time step.
- Actions are executed by the effectors and (optionally) post messages on the message list.

LCS Details (cont.)

- Periodically a reward (credit) is received from the environment. A credit allocation mechanism is used to distribute that reward to the rules, usually amongst the the chain of action sets that lead to the reward. The credit assignment affects the rule's strength.
- Periodically the GA is run on the population of rules to generate new rules and delete poorly performing ones.

GA in LCS

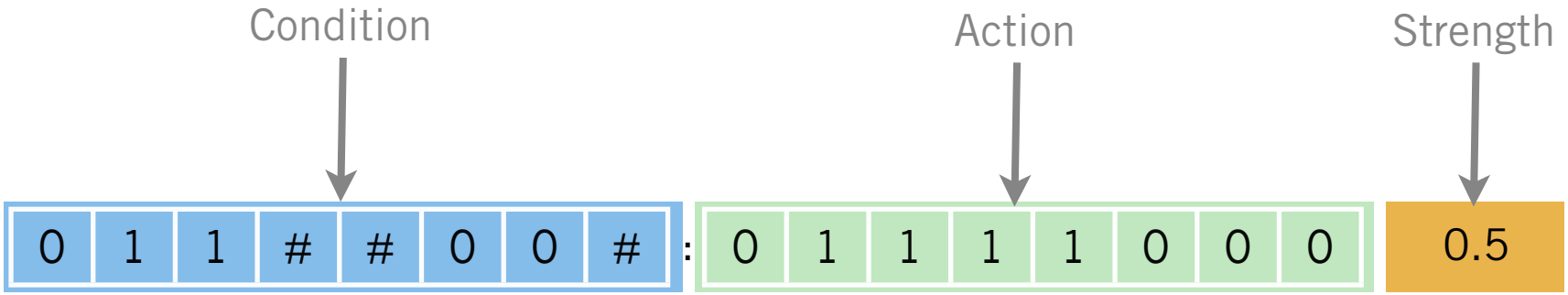
- Running the GA on rules:
  - Parent selection is based on the bid of individual rules
  - the condition:action parts of individual rules are recombined and mutated to create offspring
  - Children are given the average of their parent's strength
  - Survivor selection is based on the strength of individual rules to update the population (i.e. create a new rule set).

ZCS

- Problems with LCS:
  - LCS can suffer from **overgeneralisation** where the input space is partitioned into too few subsets.
  - Original specification was vague on certain details
- Enter **ZCS** – A "Zeroth-Level" Classifier System (Wilson 1994), a "stripped down" version of the generic LCS system
  - No internal message list
  - No explicit mechanism for transmitting information between cycles
- Covering algorithm: if no rule matches a message then create a new rule with random action that will match that condition.

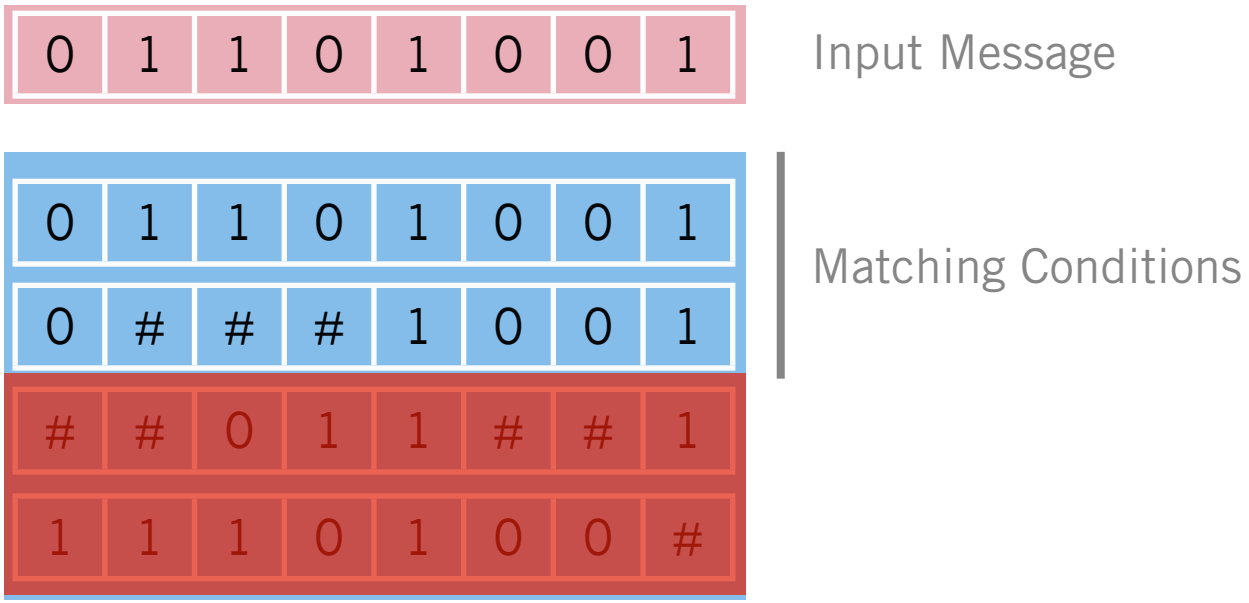
Rules

Rules represented by conditions bits, action bits and a real-valued strength: r = <c:a→ s>



Conditions drawn from the alphabet {0,1,#}, with # meaning "don't care"

For condition size L there are 2^L possible conditions.



Selection based on Rule Strength

Matching rules:

1:	0	#	1	#	#	0	#	#	:	0	1	1	1	1	0	0	0	0.5
2:	0	1	1	#	#	0	0	#	:	0	1	1	1	1	0	0	0	0.5
3:	0	1	1	0	1	0	0	#	:	0	1	1	1	1	0	0	0	0.3
4:	#	1	1	#	1	#	#	#	:	0	1	1	1	1	0	0	0	0.7

- Rule 4 has the highest strength, but is the most general
- Rules 1 & 2 have the same strength, but 2 is more specific
- The more general a rule, the more likely it will be to match an input condition ($\#^L$ matches everything!)
- A strategy is to select based on the following formula:

Specificity

$$\zeta = 1 - \frac{\text{number of \#s in condition}}{L}$$

Final Bid

$$bid = \zeta \cdot strength$$

Credit Assignment

- All rules not in the match set M^t for this cycle t are initially unchanged:

$$\forall r \notin M^t \quad s'_r = s_r$$

- All rules in the match set M^t, but not in the action set A (i.e. those advocating weaker actions) have their strengths reduced by multiplication with a factor $\tau \in [0, 1)$

$$\forall r \in M^t \setminus A^t \quad s'_r = s_r \cdot \tau$$

- All rules in the action set have a fraction of their strengths removed:

$$\forall r \in A^t \quad s'_r = (1 - \beta) \cdot s_r \qquad \beta \in [0, 1)$$

Credit Assignment (cont.)

- This strength is "pooled" and distributed equally amongst the members of the *previous* action set, after being reduced by a factor $\gamma \in [0,1)$ let $x = \sum_{r \in A^t} \beta \cdot s_r$ then:

$$\forall r \in A^{t-1} \quad s''_r = s'_r + \frac{\gamma \cdot x}{|A^{t-1}|}$$

- Finally, any feedback P^t from the system is reduced by a factor β and then distributed equally amongst the members of the current action set:

$$\forall r \in M^t \quad s'''_r = s''_r + \beta \cdot P / \mid A^t \mid$$

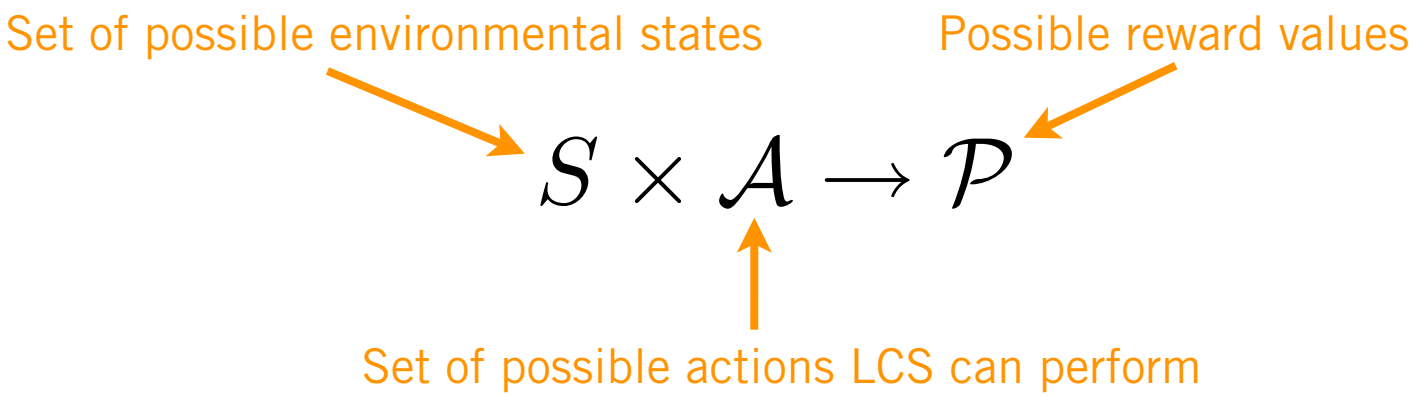
- This is the *implicit bucket brigade,* which rewards sequences of actions that lead to a reward from the environment. The discounting term Υ leads to shorter rather than longer chains of actions before rewards.

Rule Evolution

- Two rules are selected using fitness proportionate selection based on rule strength.
- One-point crossover and bitwise mutation (1,0,#)
- Offspring receives the mean of its parents' strength as initial strength.
- Offspring replace two of the current rules set, using stochastic selection inversely proportional to strength.
- "Fitness sharing" (reward sharing and fitness proportionate selection) – preserves rules sets in each niche based on relative frequency of occurrence.

Problems with ZCS

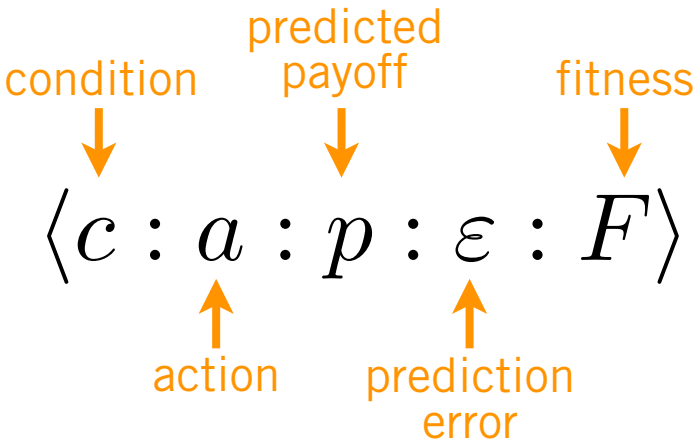
- ZCS does not preserve the complete mapping:

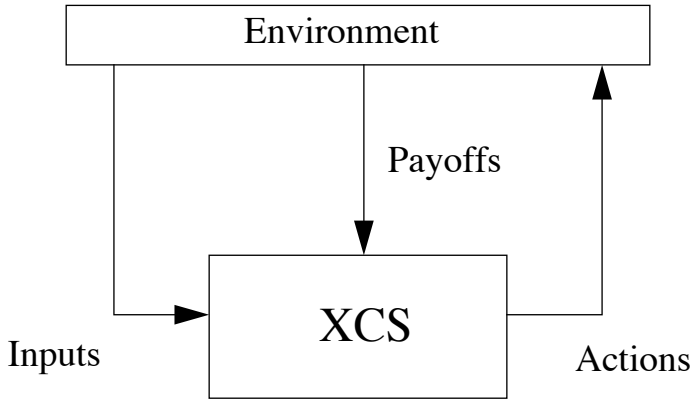


- Fitness proportionate selection may lead to premature convergence for highly fit initial solutions.
- Rule recombination from different niches may have limited value.
- Doesn't handle dynamic or noisy environments well.
- This lead to the design of XCS....

XCS

- Like ZCS, XCS has no internal message list, hence no "memory"
- Each rule is a five tuple:





- F is a prediction of fitness accuracy (rather than magnitude)
- The match set M is found as before and is logically subdivided according to actions.

Wilson, S.W. (1995) Classifier Fitness based on accuracy, *Evolutionary Computation* 3(2), pp. 149-175

XCS (cont.)

- For each action a fitness-weighted average of the rules in that subset is used as the predicted payoff:

$$p_a = \sum_{r \in R_a} w_r \cdot p_r, \quad \text{where } w_r = \frac{F_r}{\sum_{r \in R_a} F_r}$$

rule subset for action *a*

$a \in \mathcal{A}'$

set of actions present in the rues of *M*

action

- Action selection can be stochastic (e.g. selected proportional to payoff) or deterministic (select action with highest predicted payoff)

XCS (cont.)

- The action is posted to the environment. Rewards may be received from the environment. Then, rules in the *previous* action set are updated. There are 3 components:

1. Fitness values are updated using the current errors for each rule.

   To do this the accuracy is first calculated:

rule prediction error

accuracy for r → $\kappa_r = e^{\ln(\alpha)\frac{\varepsilon_r - \varepsilon_0}{\varepsilon_0}}$

system parameters

Fitness is then updated according to:

$$F'_r = F_r + \beta \cdot (\kappa'_r - F_r)$$

learning rate

relative accuracy (normalised w.r.t. the action rules in the action set)

$$\kappa'_r = \frac{\kappa_r}{\sum_{r \in R_a} \kappa_r}$$

XCS (cont.)

2. A payoff value P is calculated using the maximum predicted payoff of all actions in the current match set M (i.e., not necessarily the chosen action), discounted by a factor Υ (as per ZCS), plus any reward R received by the system in the previous time step:

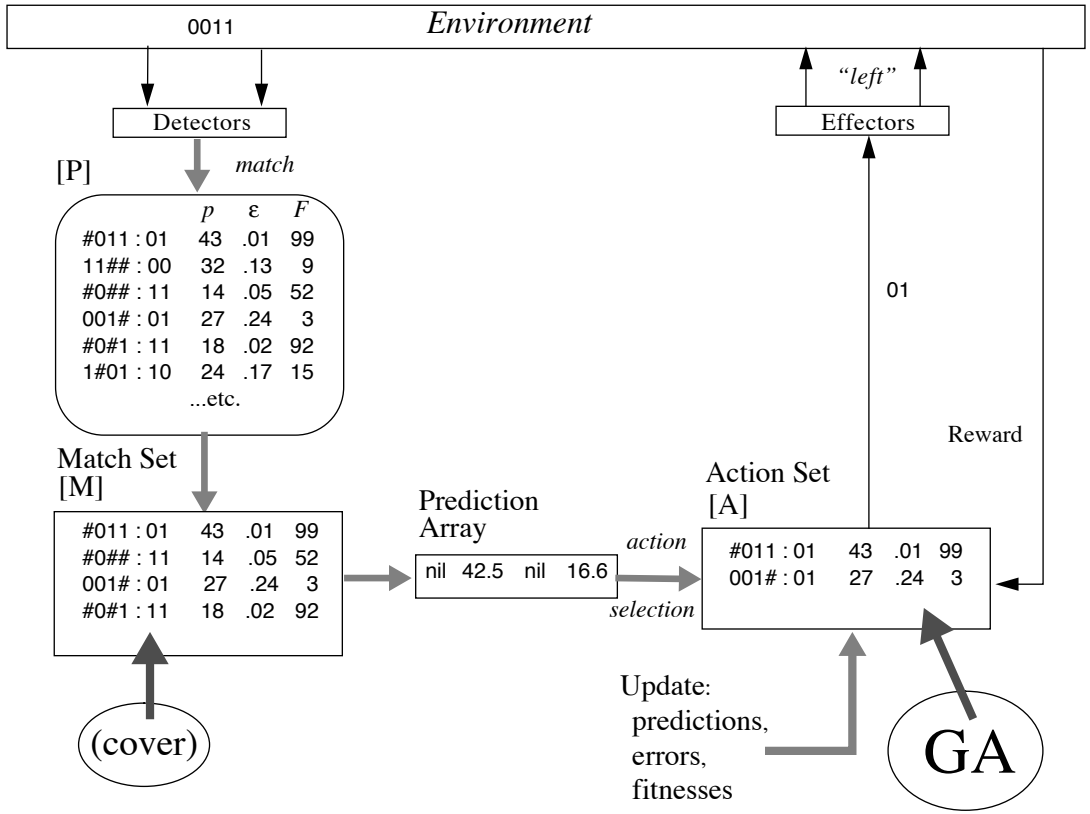
$$P = \gamma \cdot max \{p_a | a \in \mathcal{A}'\} + R$$

- P is then used to update the prediction errors:

$$\varepsilon'_r = \varepsilon_r + \beta \cdot (|P - p_r| - \varepsilon_r)$$

3. P is used to update the actual predictions p_r in the previous action set:

$$p'_r = p_r + \beta \cdot (P - p_r)$$



- For each action in [M], classifier predictions *p* are weighted by fitnesses *F* to get system's net prediction in the prediction array.
- Based on the system predictions, an action is chosen and sent to the environment.
- Some reward value is returned.

XCS (cont.)

- GA acts on the contents of the action set to achieve explicit niching
- Specialised mechanisms to control frequency of the GA running.
- Specialised mechanisms are used for initialising and managing the first few updates of p, ε and F
- P is calculated using the maximum predicted payoff of all actions in the match set (borrowed for Q-learning)
- Update rule borrowed from other areas in machine learning.
- See: http://xcslib.sourceforge.net/ for C++ version.

Woods1

```
.............................
.OOF..OOF..OOF..OOF..OOF..OOF.
.OOO..OOO..OOO..OOO..OOO..OOO.
.OOO..OOO..OOO..OOO..OOO..OOO.
.............................
.............................
.OOF..OOF..OOF..OOF..OOF..OOF.
.OOO..OOO..OOO*.OOO..OOO..OOO.
.OOO..OOO..OOO..OOO..OOO..OOO.
.............................
.............................
.OOF..OOF..OOF..OOF..OOF..OOF.
.OOO..OOO..OOO..OOO..OOO..OOO.
.OOO..OOO..OOO..OOO..OOO..OOO.
.............................
```

Wilson's *Woods1* environment:
O = rocks
F = food
. = empty space
* = agents current position

Woods1 - results (ZCS)

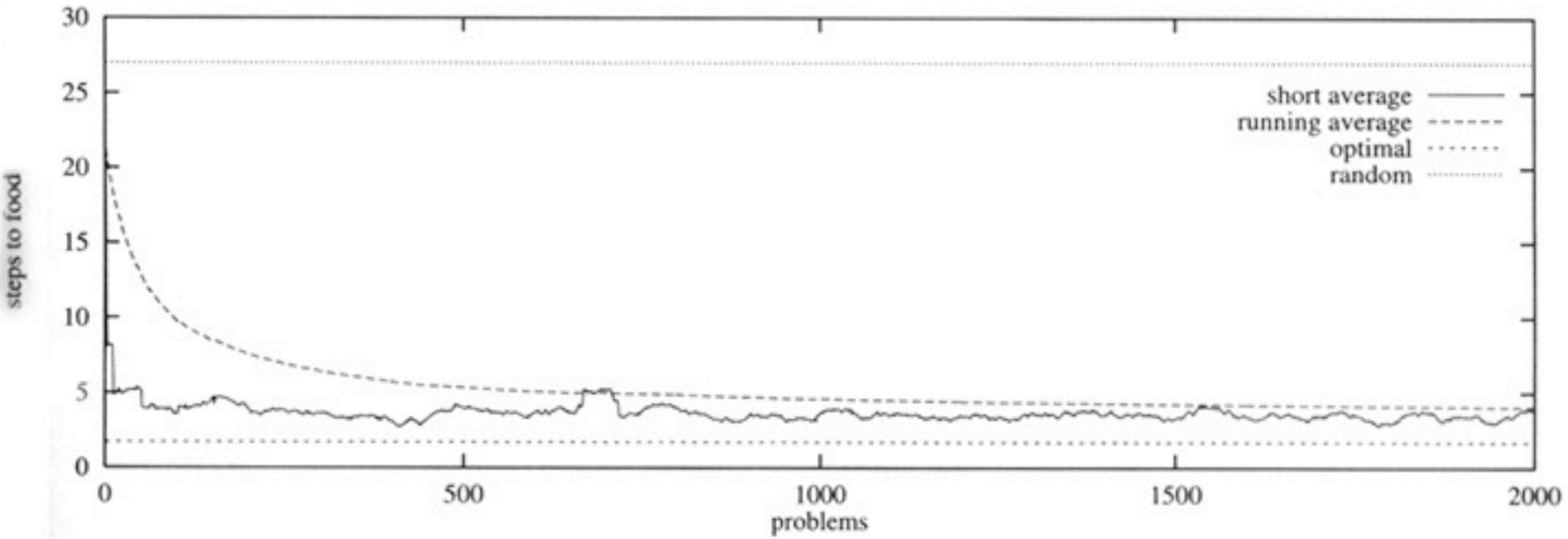
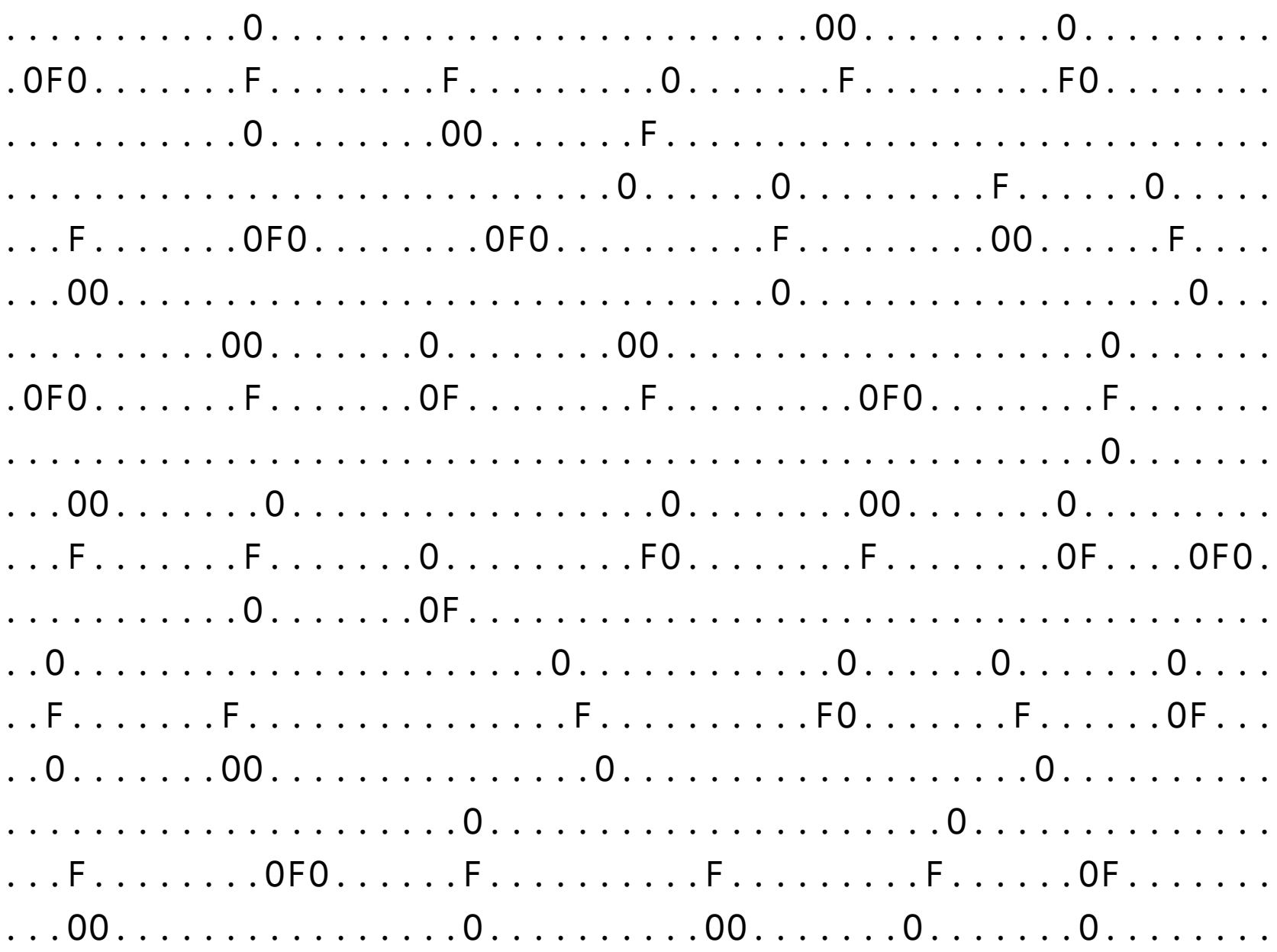


Figure 21.5 ZCS performance in Woods1: The "short average" plot is an average over the 50 most recent trials. The running average curve is over all past trials.

Woods7



Woods7 - Results (ZCS)

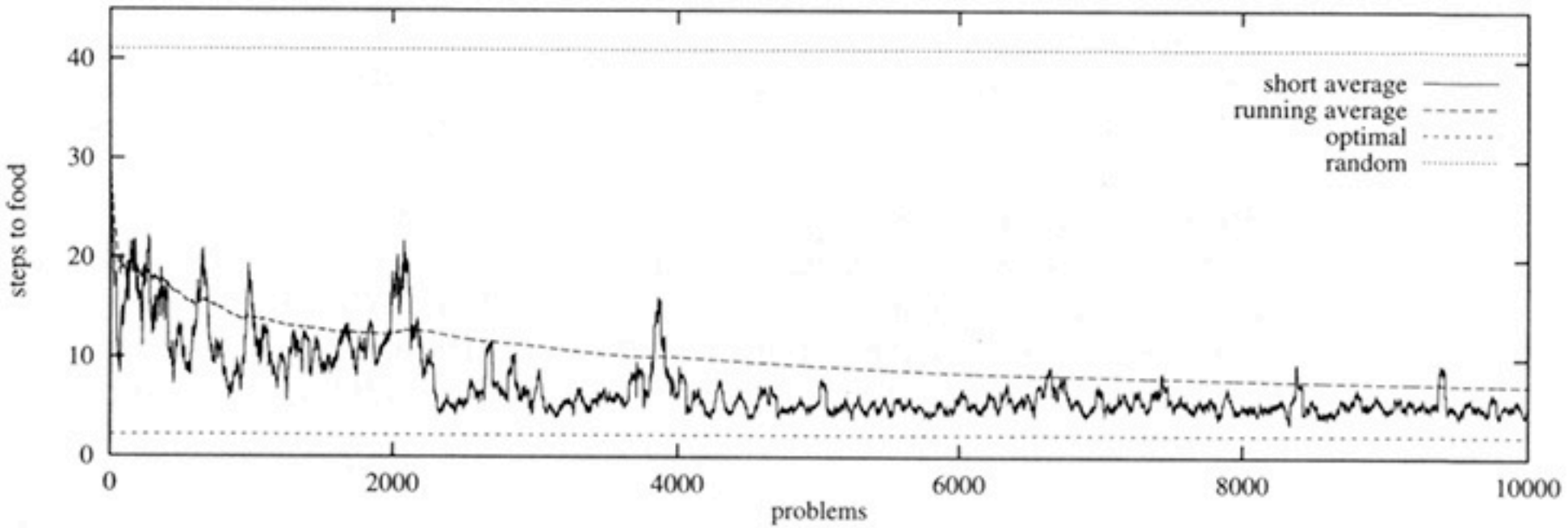


Figure 21.7 ZCS performance in Woods7: The "short average" plot is an average over the 50 most recent trials. The running average curve is over all past trials.