Classifier Conditions Using GEP

—or—

Can We Have Conditions that Fit?

Stewart W. Wilson
Prediction Dynamics, Concord, MA
Department of Industrial and Enterprise Systems Engineering,
The University of Illinois at Urbana-Champaign IL 61801 USA

What is a Condition?

$$< condition > : < action > \Rightarrow < prediction >$$

—defines a subset of the input space, a generalization over a region with the same payoff or related payoffs

Examples of Condition Syntax

$< 1, 0, \# >$	Binary conjunct
$< l_0, u_0, ..., l_n, u_n >$	Hyperrectangle
$< hyperellipsoid >$	Orientable, n-d ellipsoid
$< convex\ hull >$	Arbitrary convex region
$< neural\ net >$	Arbitrary region
$< function >$	Arbitrary region

Why Should the Condition Fit?

—Minimize number of classifiers needed

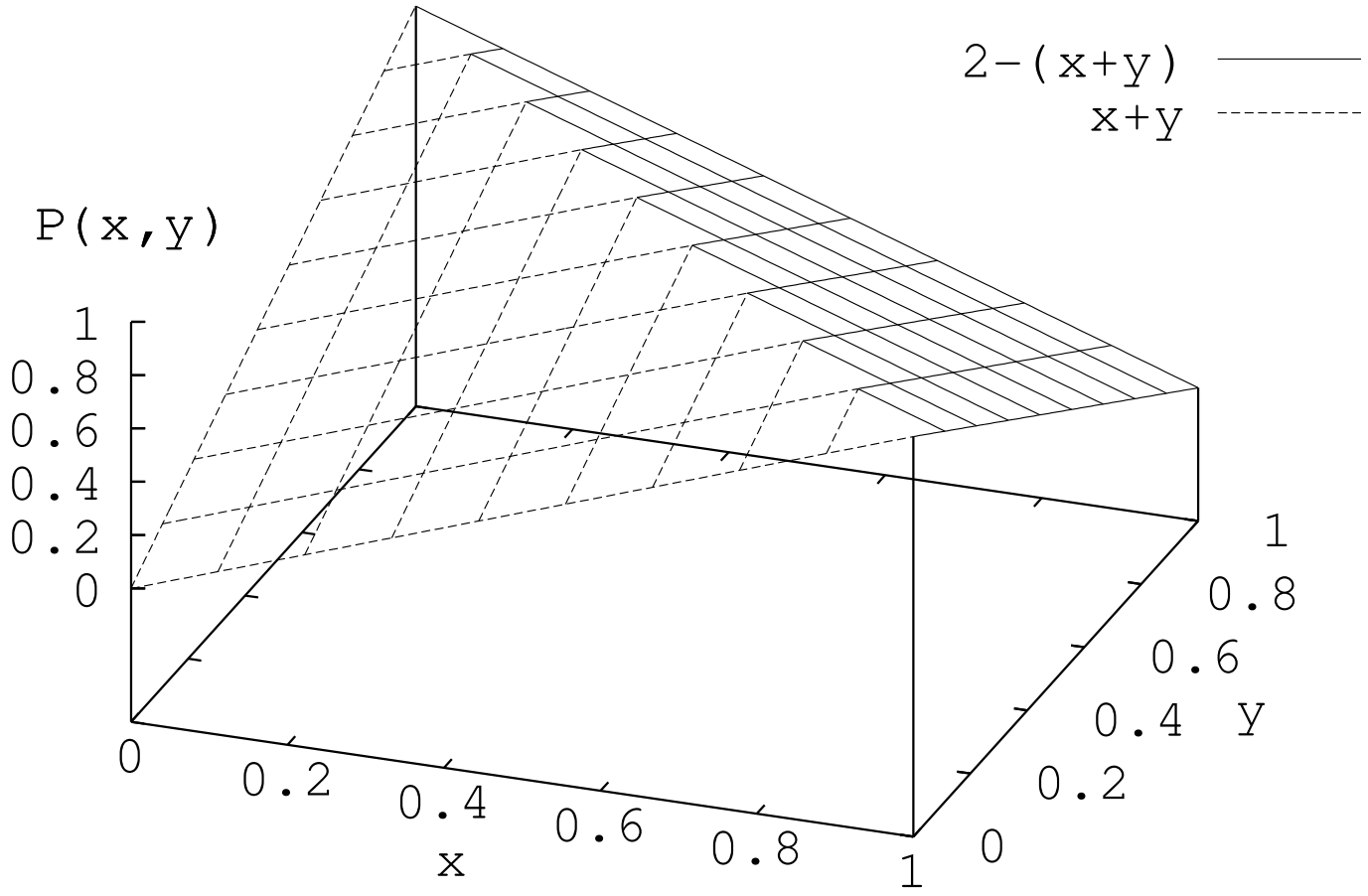
—Transparency of system's knowledge

Why Functional Conditions?

—Exact fit!

—See the knowledge (vs. NN)

Example Payoff Landscape



How Will You Do Functional Conditions?

—Gene expression programming (GEP) (C. Ferreira)

—May work better than GP

What is the Big Deal with GEP?

—**Genome to phenotype translation**

—Every chromosome/expression-tree is valid

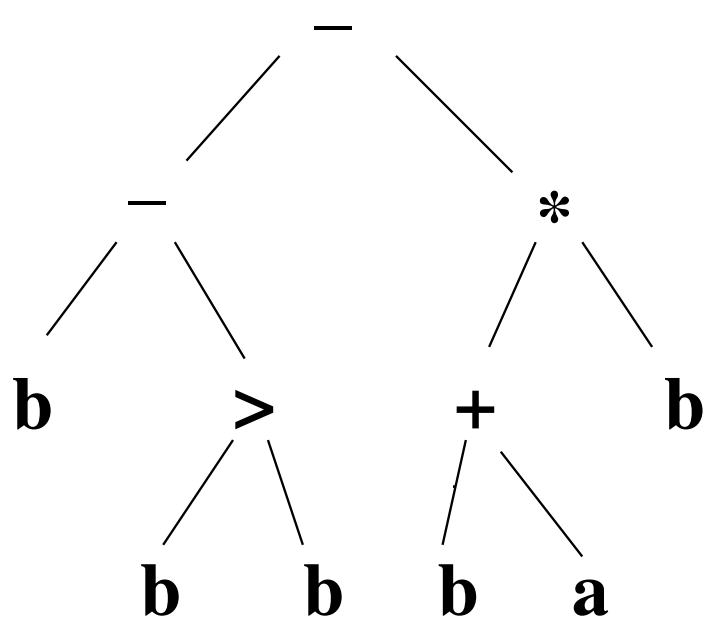
—Permits arbitrary genetic operators

—"Bloat" is limited

Example of Translation

Chromosome: `( - - * b > + b b b b a b b )`

Expression Tree:



Algebraic: $b - (b + a)b$

How Do You Do Matching?

—Use threshold (Bull)

$$b - (b + a)b > 0$$

$$1 - (b + a) > 0$$

$$b + a < 1$$

XCSF-GEP

—Like XCSF

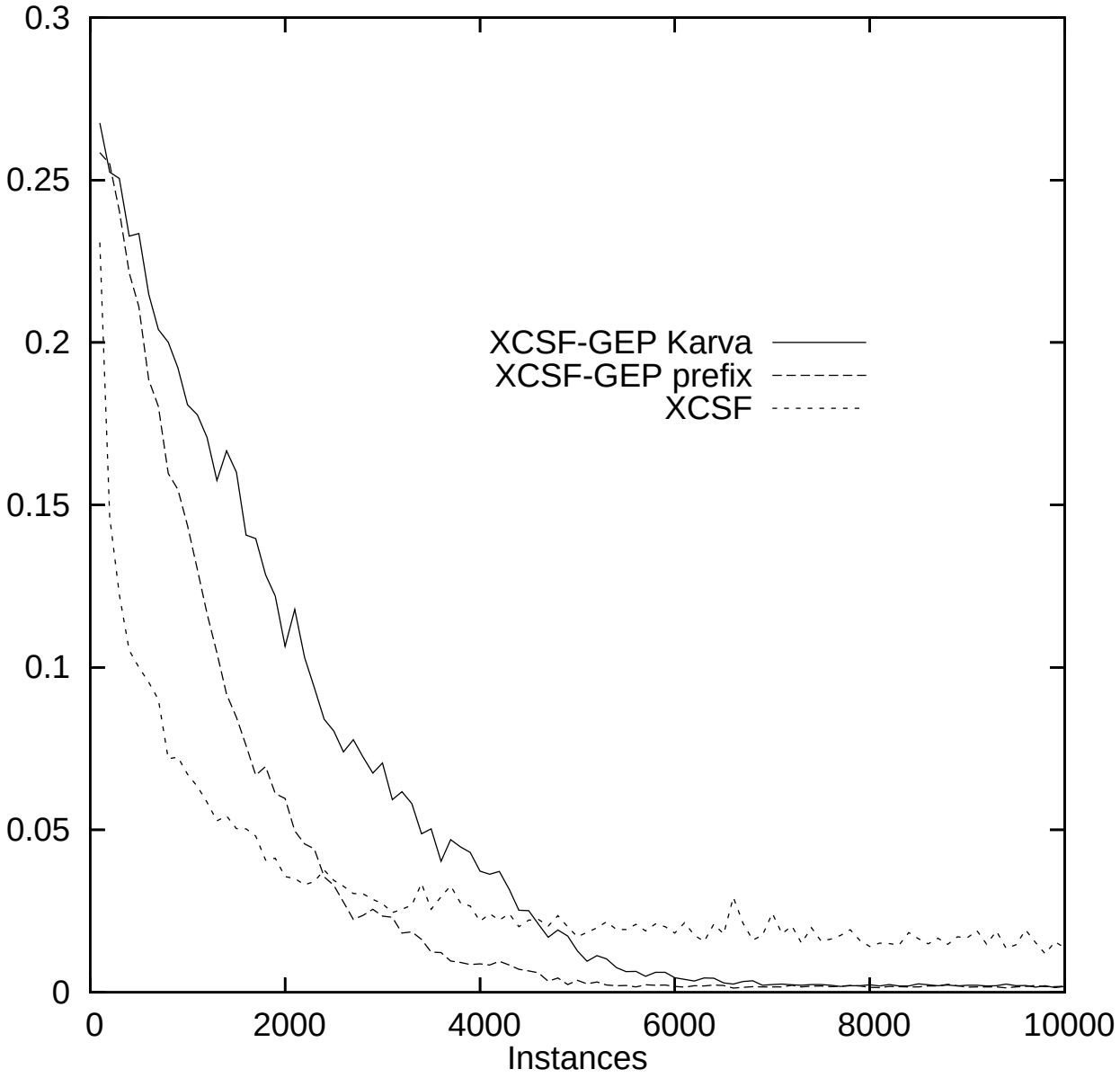
—Condition is a chromosome

—Expression tree evaluated and compared with threshold

—Discovery component applies genetic ops to chromosome

—Covering by random chromosome generation until one matches

Results on Payoff Landscape



Example Evolved Classifiers

	Chromosome	Algebraic equivalent	Domain
1.	`(- - * b > + b b b b a b b)`	$b-(b+a)b$	$a+b<1$
2.	`(* - * + / b b a b b b a b)`	$((a+b)-1)b^2$	$a+b>1$
3.	`(> - - / b + a a a a a b a)`	$(1-b)>a$	$a+b<1$
4.	`(> b - > a / b b a b a b a)`	$b>((b/a>b)-a)$	$a+b>1$
5.	`(> - * a / / b b a b a b a)`	$(a(1/a)-b)>a$	$a+b<1$
6.	`(- * * a / + b a b b a b a)`	$a((b+a)/b)b-a$	$a+b>1$
7.	`(- * a + * + b a b a a b a)`	$((b+a)b+a)a-a$	$a+b>1$
8.	`(- - > / / + a b b b b b a)`	$((a+b)/b^2>b)-b-a$	$a+b<1$

What about Compactness?

—Apply Compact Ruleset Algorithm (CRA) to final population

—Result: exactly 2 classifiers covering all inputs

Example of Compact Ruleset

(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 0 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 1)
(0 0 0 0 0 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 1 1)
(0 0 0 0 0 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 1 1 1)
(0 0 0 0 0 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 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 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 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)

(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 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 1 1 1 1 1 0 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 1 1 1 1 1 0 0 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 1 1 1 1 1 0 0 0 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 1 1 1 1 1 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

How Do You Do Constants?

—Use Koza's "Ephemeral random constant" (ERC) with mutation

—Add ERC to terminal set

Results with a Different Diagonal

```
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 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 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
```

```
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1)
(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 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
```

```
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
```

Results with Circular "Mesa"

(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1)
(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 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1)

(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0)
(1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0)

Can It Do Hard Problems?

—Not Sure **:)**

—Closed regions seem to be difficult

—Need generality pressure

Thank You.