マルコフ連鎖モンテカルロ法によるソフトウェアテストケースの設計

周搏，岡村寛之，土肥正
広島大学大学院工学研究科

自己紹介

所属

広島大学大学院工学研究科情報工学専攻
ソフトウェア信頼性工学講座
システム信頼性工学研究科目（研究室）

専門分野

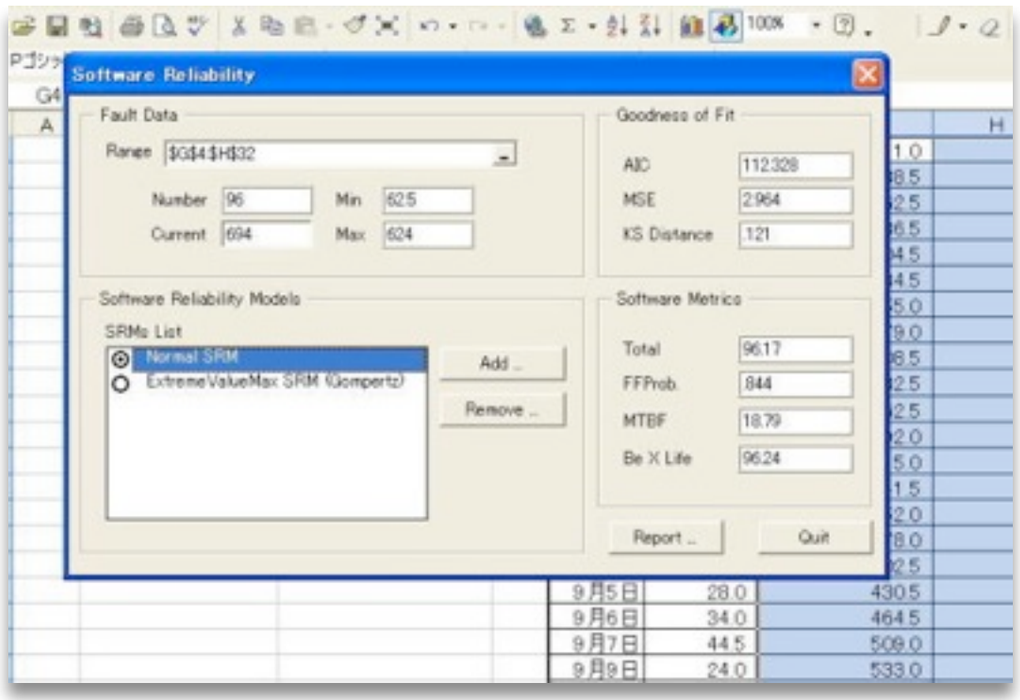
ソフトウェア信頼性工学
確率統計によるシステムの性能評価

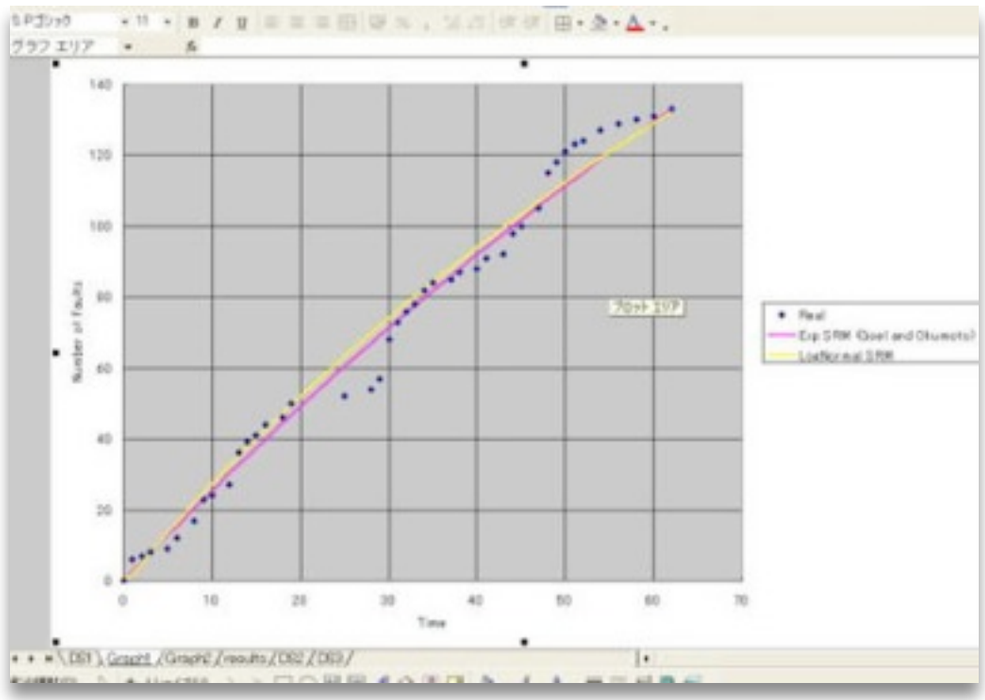
信頼性評価ツール

SRATS: Software Reliability Assessment Tool on Spreadsheet Software

Microsoft Excel で動くソフトウェア信頼性評価ツール

http://www.rel.hiroshima-u.ac.jp/~okamu/SRATS/manual/home.html





発表概要

背景

テストケース設計と確率・統計

ベイズ統計を応用したテストケース設計

モデル化

MCMCによる手法

数値例

ソフトウェアテスト技法

- Black Box Testing
  - 同値分割
  - 限界値分析
- White Box Testing
  - 命令網羅
  - 判定条件網羅
  - 条件網羅

→ なぜ良いか？理論的な裏付け

経験則の本質

プログラム処理の境界にバグあることが多い

代表値だけで大丈夫だ

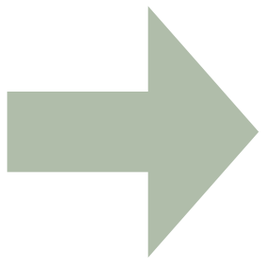
各命令は最低１回は実行しておくべきだ

バグはたった２つの入力の組み合わせが
で原因になってることが多い

経験的に獲得した（学習した）知識
何かしらの形でモデルを形成し学習した

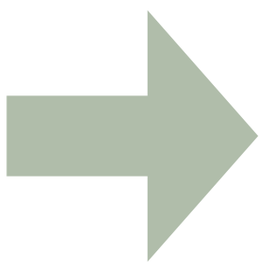
発表の目的

経験則を確率・統計的に扱うモデルの考察



テストケース設計を統一的に議論する数学的な基礎

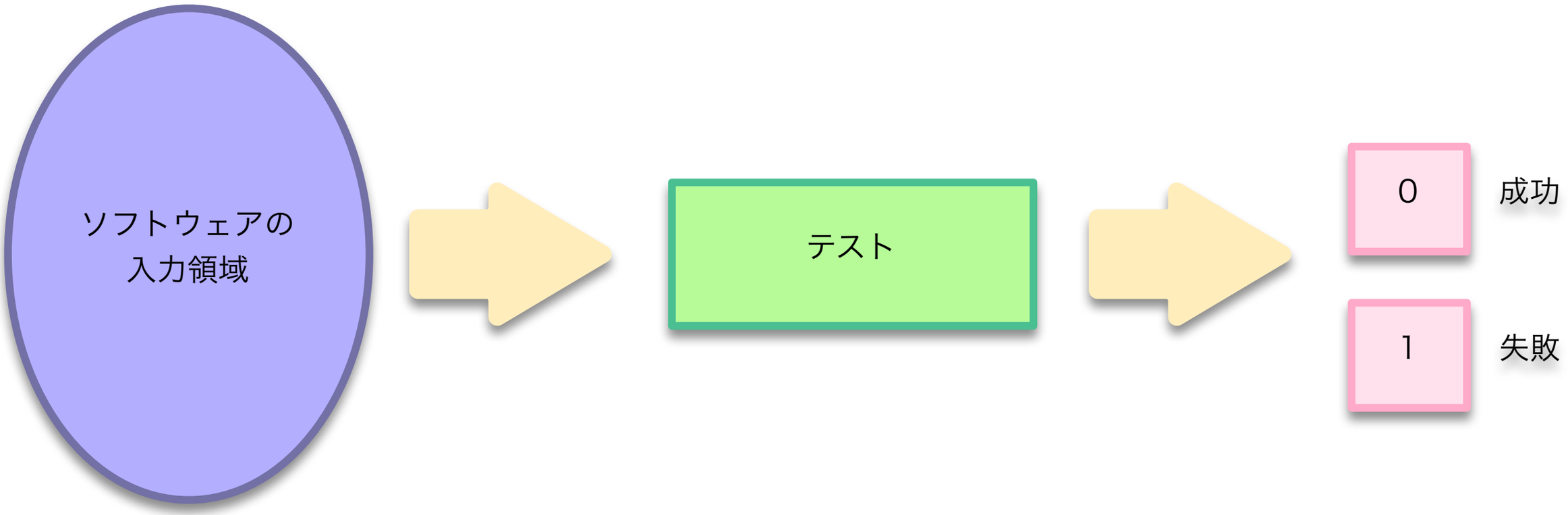
既存の統計手法をモデルに適用したテストケース設計手法の考察



マルコフ連鎖モンテカルロ法によるテストケース設計の試み

ソフトウェアテストモデル

ソフトウェアテストを数学的に抽象化



テスト関数の出力が１となる領域を効率的に探す

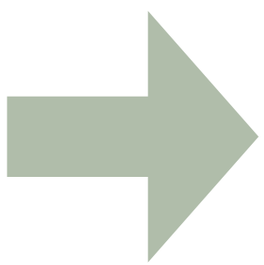
手がかりは入力領域の特徴

Contd.

例えば．．．

同じ結果を出力する（と思われる）領域の代表値をテストする

命令を１回ずつ実行する領域の出力が０ならばそれ以外の領域も０である（と考える）



プログラムの構造と経験則

統計的にもっと単純化

Contd.

統計モデルとして．．．

同値クラス：同じ結果を出力する確率が高い入力
（出力の相関が強い）

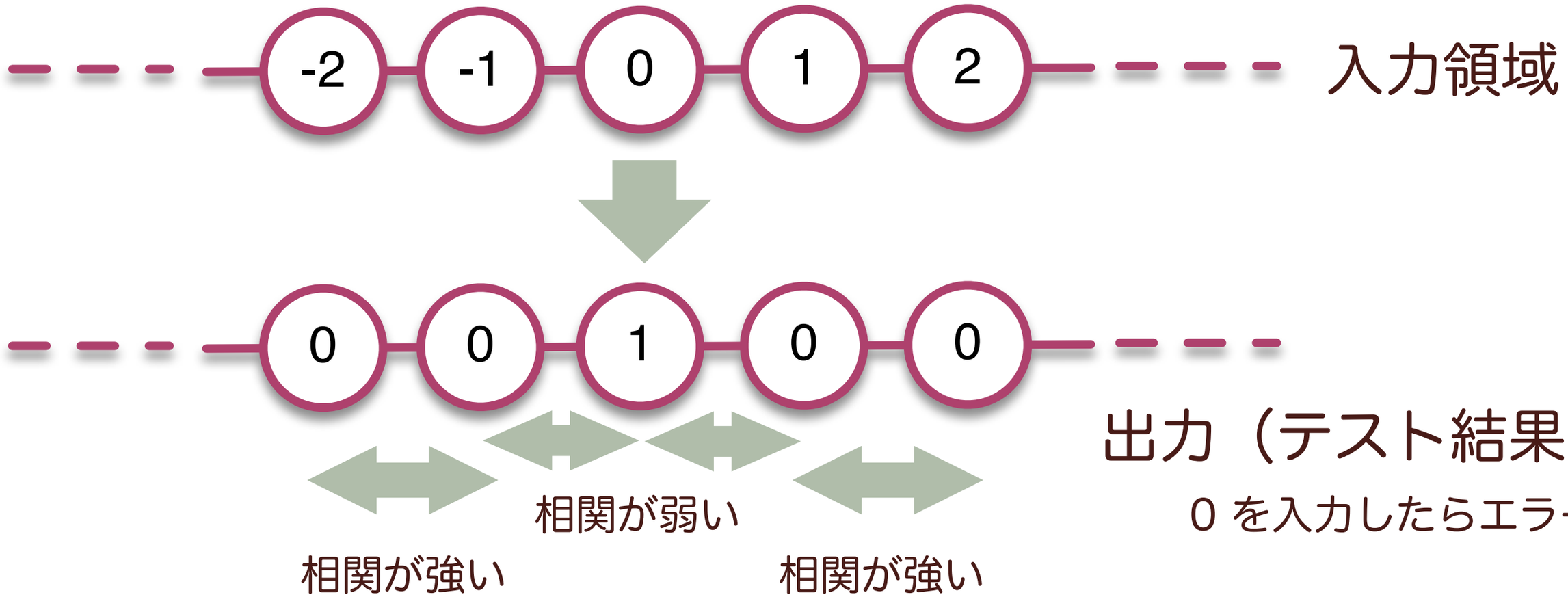
命令を１回ずつ実行する領域の出力が０ならばそれ以外の領域も０である確率が高い（領域間の相関が強い）

入力間の出力に対する相関の強弱を
モデル化

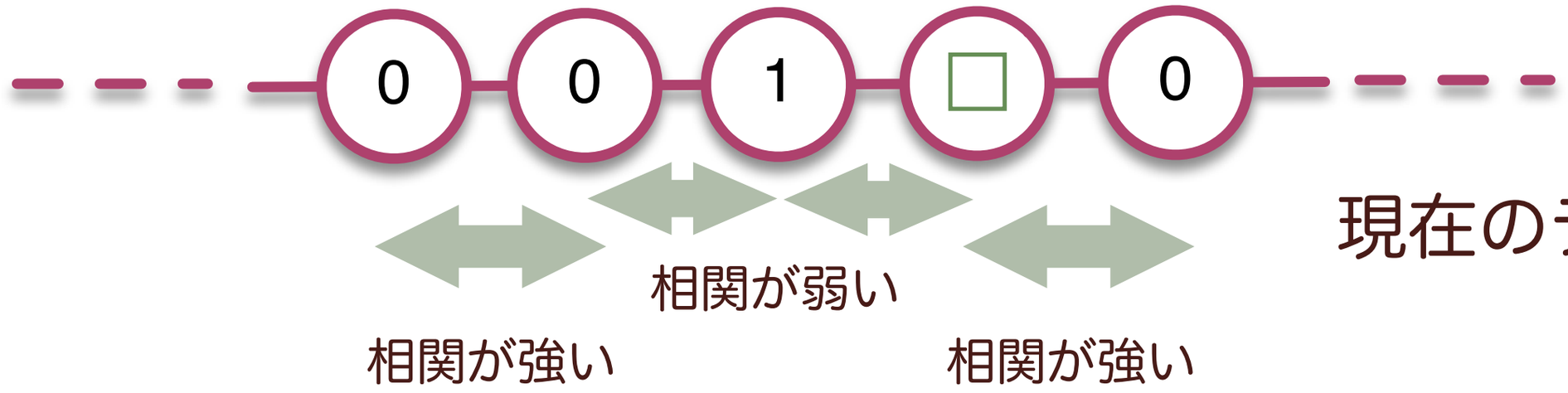
例

ある int 型の引数をもつ関数

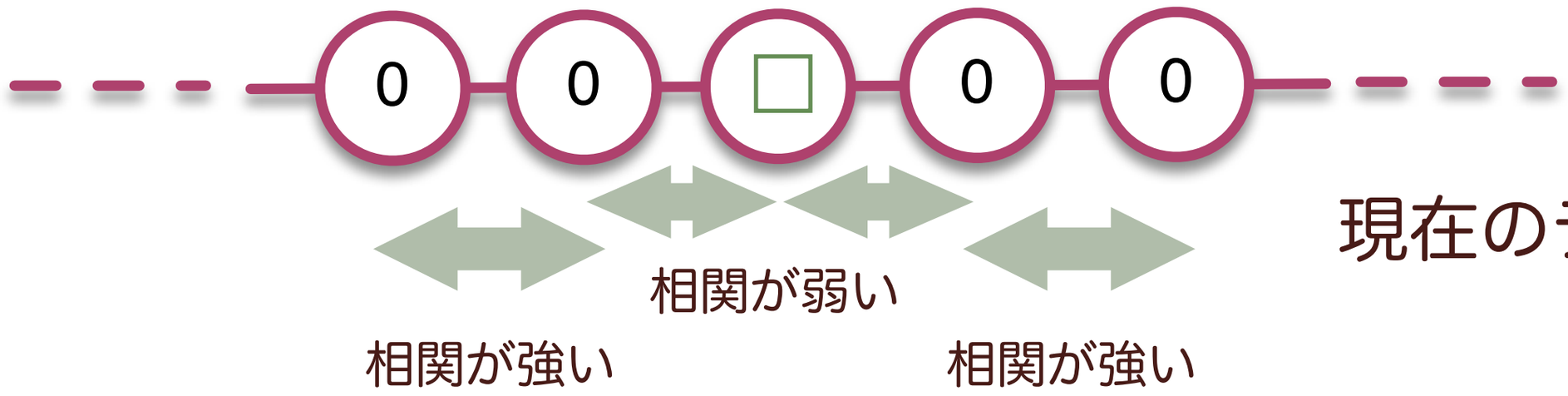
```
void func(int x)
```



Contd.

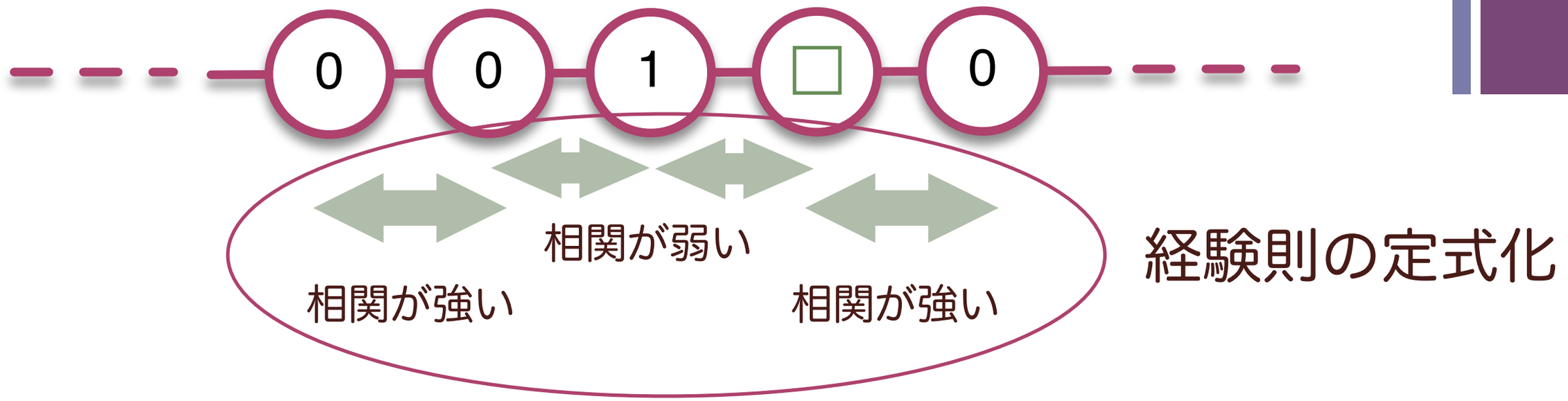


□ 部分のテストをする必要はない

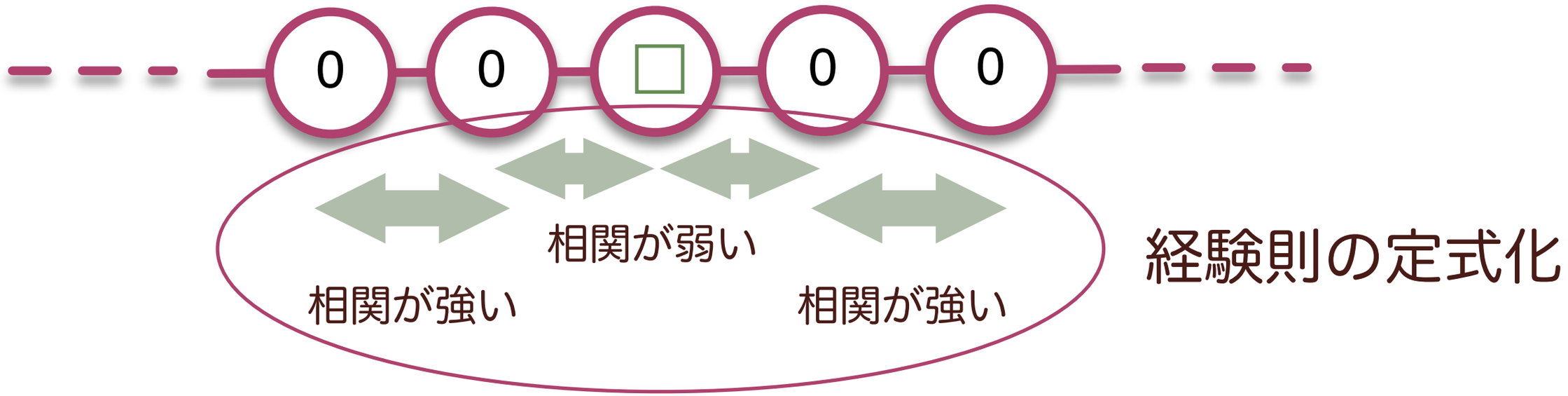


□ 部分のテストをする必要がある

Contd.



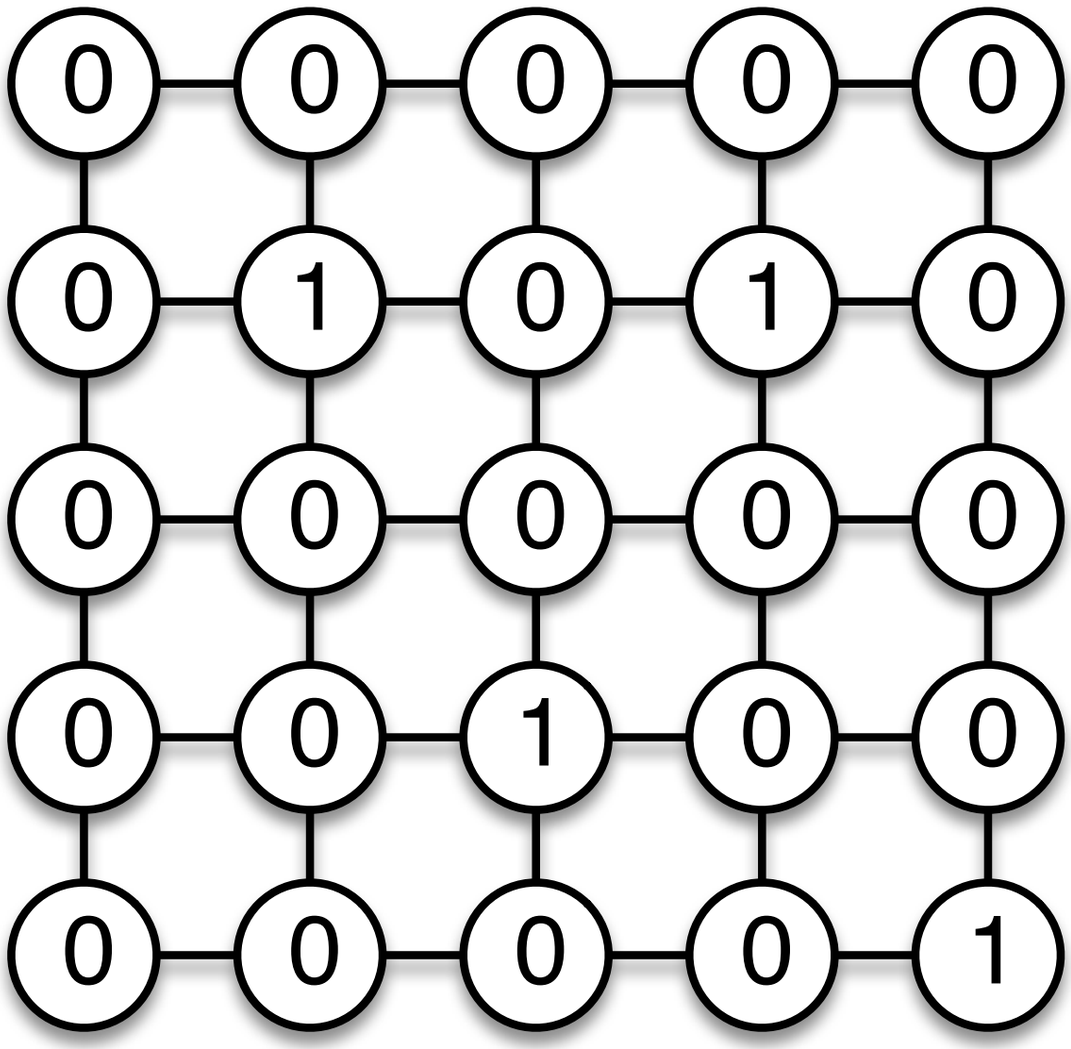
□ 部分のテストをする必要はない



□ 部分のテストをする必要がある

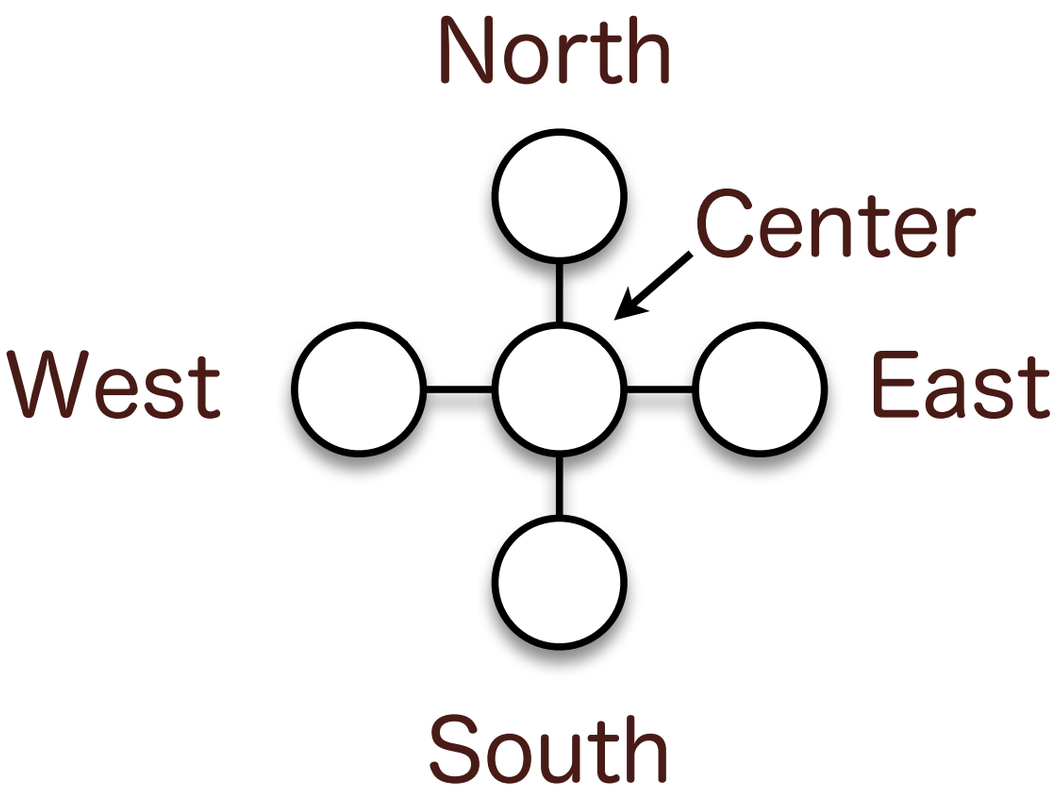
テストケースモデル

２次元格子　`void func(int x, int y)`



Contd.

経験則：周辺のテスト結果に対する条件付き確率



Center\North	success	failure
success	0.8	0.2
failure	0.3	0.7

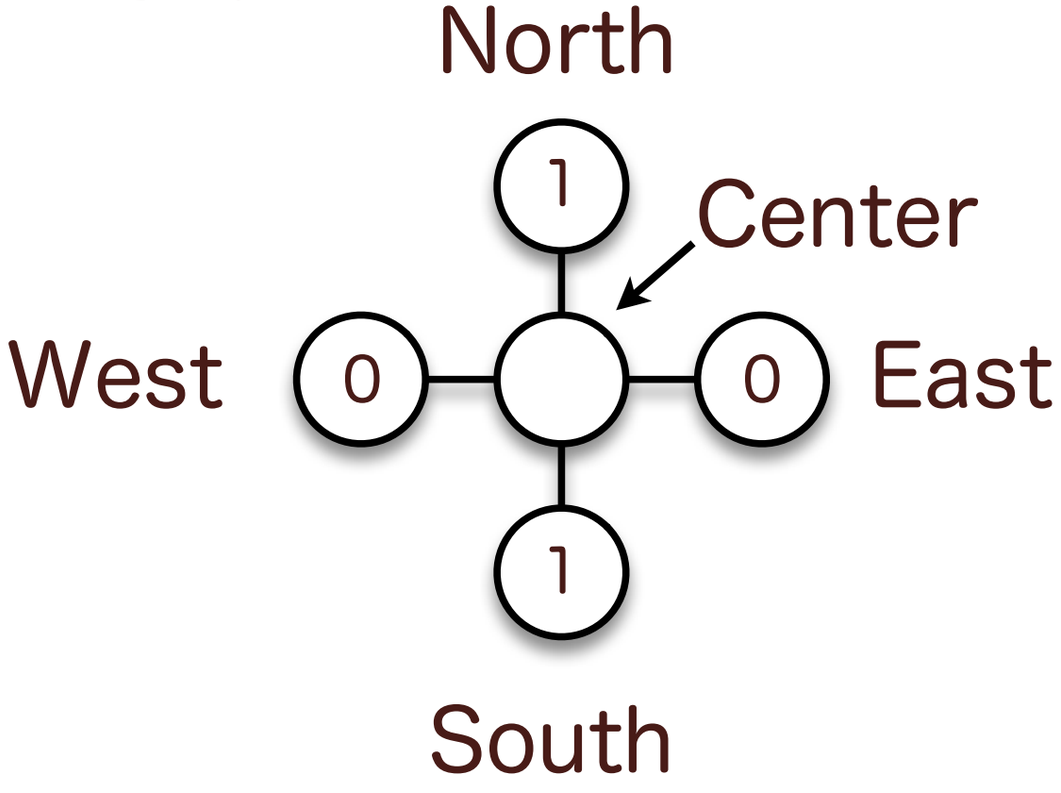
Center\West	success	failure
success	0.8	0.2
failure	0.5	0.5

Center\South	success	failure
success	0.8	0.2
failure	0.2	0.8

Center\East	success	failure
success	1.0	0.0
failure	0.0	1.0

ベイズ推定

ベイズ推定により周辺（North, West, East, South）のテスト結果から中心（Center）の結果を推測



F_N, F_S: North, South にバグがある

S_W, S_E: West, East にバグがない

F_C: Center にバグがある

$$P(F_C|F_N, S_W, S_E, F_S) = \frac{P(F_N|F_C)P(S_W|F_C)P(S_E|F_C)P(F_S|F_C)P(F_C)}{Z}$$

$$Z = P(F_N|F_C)P(S_W|F_C)P(S_E|F_C)P(F_S|F_C)P(F_C) + P(F_N|S_C)P(S_W|S_C)P(S_E|S_C)P(F_S|S_C)P(S_C)$$

MCMC

サンプリングを用いてベイズの事後確率を求める一手法

テストケース設計への応用

テストの途中経過，経験則（事前分布，条件付き確率）から各入力にバグが潜んでいる確率を算出する

テスト結果をフィードバック

バグが潜んでいる確率が高い入力を優先してテストする

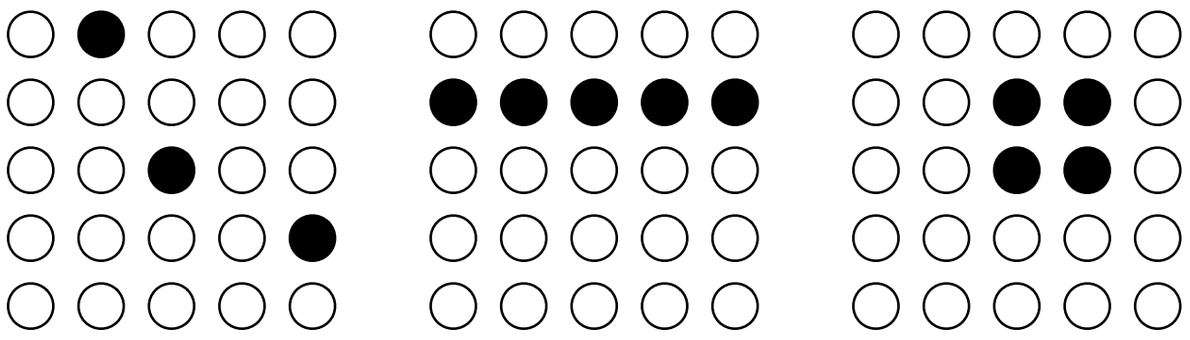
数値例

簡単なテストに関するモデル
＋MCMCによる入力の選択

２次元格子（20×20）の入力領域

障害シナリオ

Point: 無作為にバグ
Strip: 線状に相関を持つバグ
Block: 長方形に相関を持つバグ



(point)　(strip)　(block)

相関

各入力から４方向（North, South, West, East）に隣接した入力に対して次の条件付き確率

Center\NWES	success	failure
success	0.9	0.1
failure	0.1	0.9

$p = 0.9$

Center\NWES	success	failure
success	0.8	0.2
failure	0.2	0.8

$p = 0.8$

Center\NWES	success	failure
success	0.7	0.3
failure	0.3	0.7

$p = 0.7$

結果

バグが潜んでいる確率の高い入力からテスト
通常のランダムテストと比較

評価尺度

最初のバグが発見されるまでのテスト回数

		Point	Strip	Block
RT		19.4 (301.7)	19.5 (270.1)	19.8 (304.1)
MCMC-RT	$p = 0.9$	18.1 (298.4)	18.7 (293.0)	18.6 (263.5)
	$p = 0.8$	17.9 (279.6)	18.5 (284.0)	18.2 (249.2)
	$p = 0.7$	17.3 (253.0)	18.0 (253.0)	17.0 (233.9)

まとめ

ソフトウェアテストの統計からのモデル化

テストアクティビティのモデル化

ベイズ統計を用いたテスト設計アプローチ

今後の課題

実プログラムおよび経験則の関連