SPINによるモデル検査（1）

タオベアーズ合同会社
のなかあきら

内容

- SPIN概要
- Promela言語概説
- デモ

SPINの紹介

- 最も広く使われている、オープンソースの並列、分散処理ソフトウエア検証ツール
- 1980年 Bell LabのHolzmann博士らが電話交換機制御ソフト検証ツールとして開発着手
- 初期の目的はプロトコル検証
- 2001年ACM Softwareアワード受賞(他にはUnix,Java,TCP/IP,Tcl/Tk等)

SPINの紹介

- SPINをエンジンとした派生ツールが多い
- 現在でも活発に開発が進行している
- SPIN(Simple Promela Interpreter)
  - ツールの名前
- PROMELA(Process Meta Language)
  - モデル記述言語の名前

SPINの特徴

- 並列処理、マルチスレッドプログラムの論理的・機能的な欠陥を設計段階で発見する
- ソフトウエア専用のモデル検査ツール

SPINの歴史

- 1991年 1.0 Initial version
- 1995年 2.0 Partial order reduction
- 1997年 3.0 Minimised automaton representation
- 2003年 4.0 Embedded C code
- 2005年 5.0 Multicore support
- 2008年 5.1.5 Swarm support

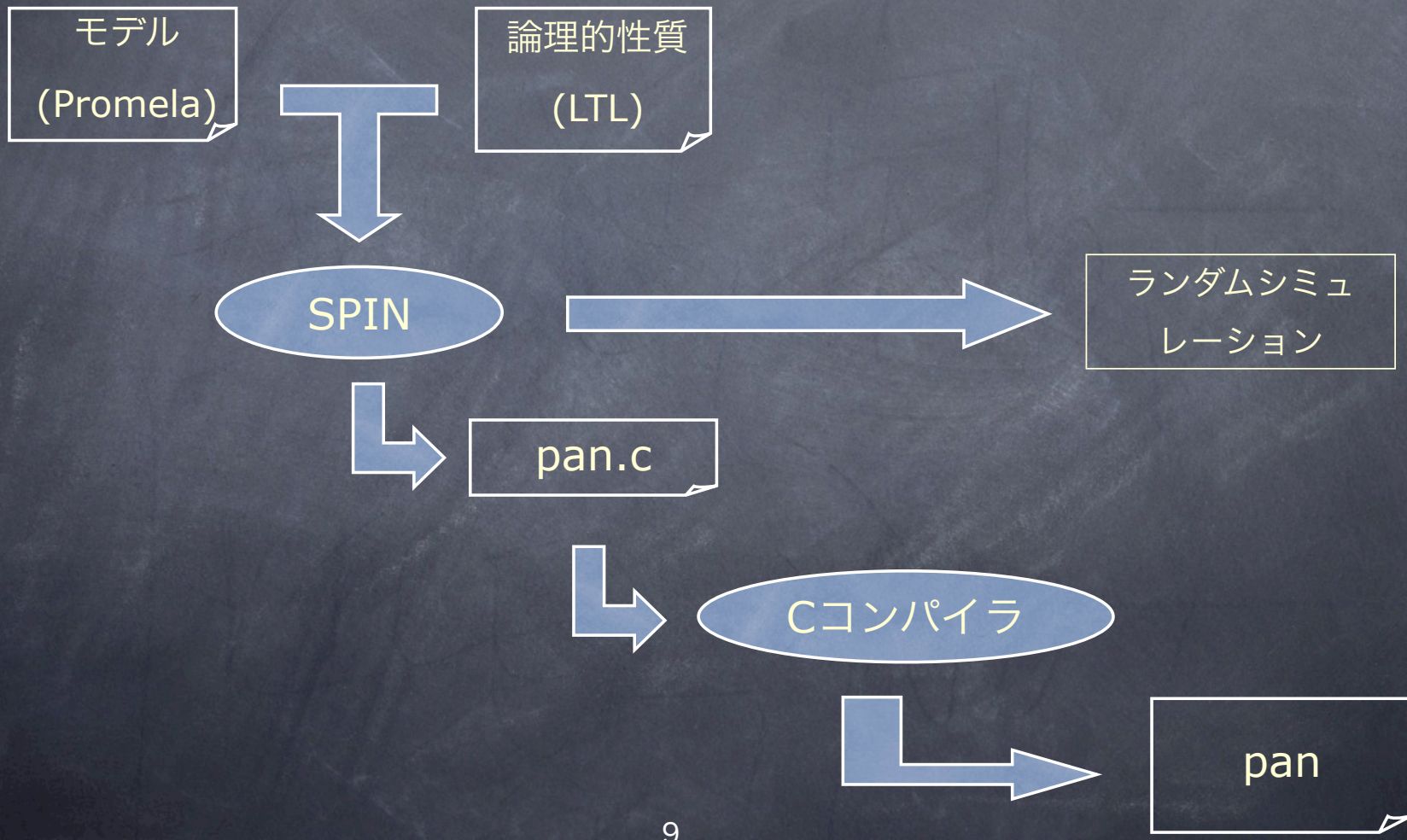
プラットフォーム

- Windows(Cygwin),Mac,Unix, etc.
- ANSI-C　コンパイラが必要
- XSPIN (GUI環境)　　（Tcl/Tk が必要）
- 統合環境 jSpin　（Javaが必要）

SPINの動作モード

- シミュレーションモード
  - 各プロセスをランダムなタイミングで実行する
- 検証モード
  - あらゆる状態遷移を生成し、所定の性質を満たしているか検証する

処理の流れ



Promela言語概説

３種類のオブジェクト

- プロセス
- ローカルデータとグローバルデータ
- メッセージチャネル

プロセス

- proctypeでプロセスの型を定義
- プロセスはproctypeがインスタンス化されたもの
- active または run でインスタンス化
- 他のプロセスとの通信はグローバル変数またはチャネルを使う

まずは Hello World

```
active proctype helloworld(){
  printf("Hello World!\n");
}
```

データオブジェクト

- bit, bool
- byte
- chan, mtype
- pid, short, int
- unsigned

データオブジェクト

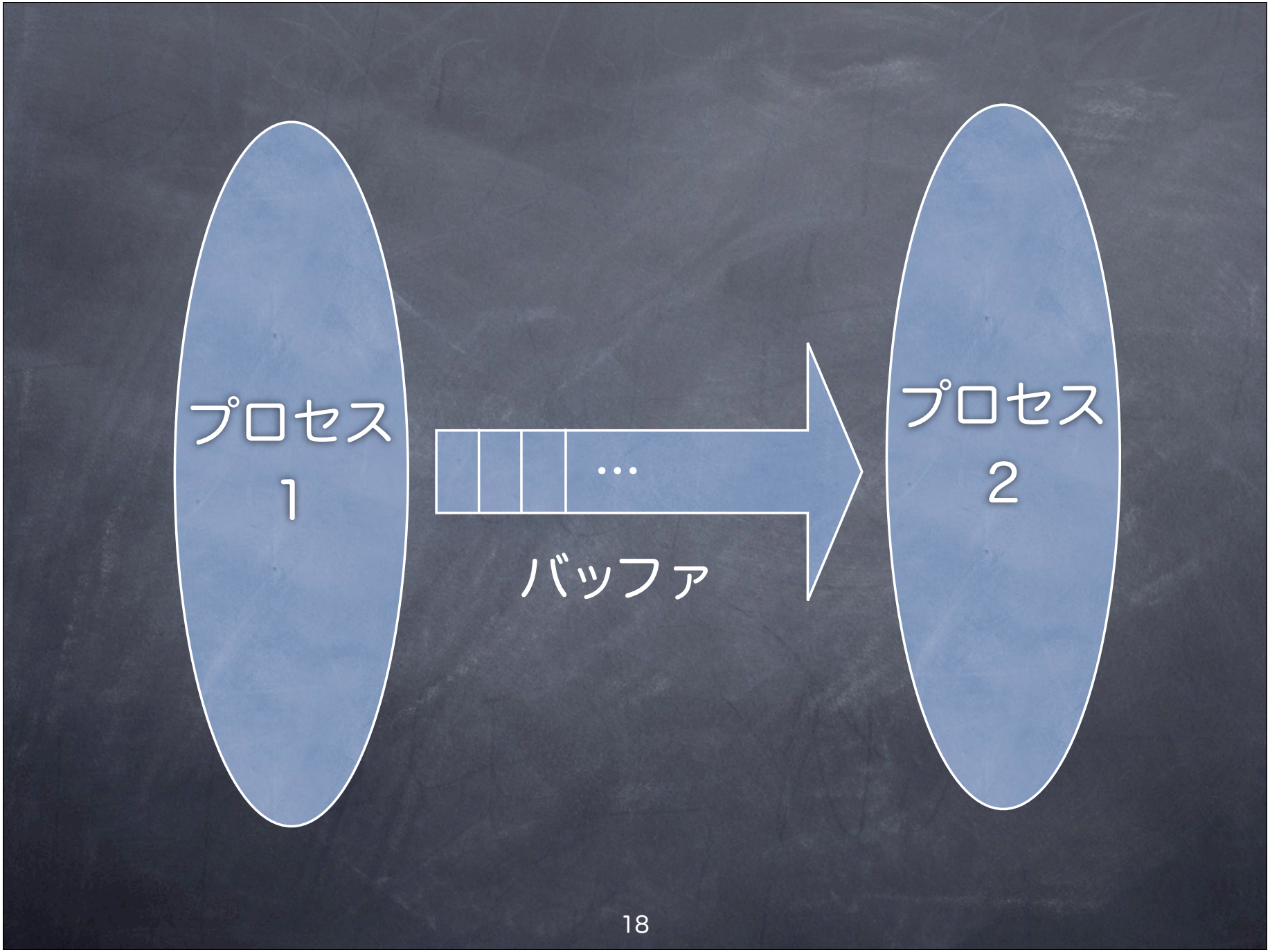
- 文字列型、浮動小数点型は無い

データオブジェクト

```
byte x = 100;        グローバル変数
bool b = false;
mtype = { red, yellow, green }; 列挙型
active proctype foo(){
    int z = 10000;    ローカル変数
    mtype light;
    b = true;
    light = green
}
```

チャネル

- プロセス間通信に使用する通信路
- 送信、受信操作が結び付けられたデータ型
- chan ch = [バッファサイズ] of {型;...}
- サイズが０の時はランデブーチャネルとなる


プロセス
1
…
バッファ
プロセス
2

チャネル

```
chan c = [10] of {int};
active proctype snd(){
  int x = 77;
  c ! x
};
active proctype rcv(){
  int y;
  c ? y;
  printf("Received: %d\n",y)
}
```

制御構造

選択

```
if
  :: ガード -> 本体；
  :: ガード -> 本体；
fi
```

繰り返し

```
do
   :: ガード -> 本体；
   :: ガード -> 本体；
od
```

言語の４つのコンセプト

- 有限状態
- 非同期的実行
- 非決定性
- 実行可能性

有限な状態

- 変数の大きさ、チャネルのバッファの大きさ、プロセス数（255）,etc.
- モデルは閉じてなければいけない（入力はない）
- 検証のためには外部環境もモデル化する

非同期的な実行

- プロセスは任意のタイミングでインターリーブされる

制御構造と非決定性

非決定性

- 一つでも真のガードがあれば、if,doは実行される
- 全てのガードが偽ならば、if,do全体がブロックされる
- 一つ以上ガードが真の場合、どれか一つの文がランダムに選ばれ実行される

非決定性構文によるモデリング

- 非決定的な事象のモデリングにとても便利
- 例
  - 時々パケットが消失する
  - 時々ユーザーが変なボタンを押す
  - 時々ハードウエアのエラーが起きる

Promelaの非決定性構文

```
if
  :: a => 100 -> b = b + 1;
  :: a =< 100 -> b = b - 1;
fi
```

ランダムな動作の実行

- nに1,2,3のいずれかを代入する

```
if
    :: n = 1;
    :: n = 2;
    :: n = 3;
fi
```

実行可能性

- 全ての文は実行可能な場合とブロックされる場合がある
- 式は偽ならばブロック、真ならば実行可能
- 代入は、常に無条件に実行可能

実行可能性

x = 1;　無条件に実行される

x > 2;　真となるまでブロックされる

y = 3;　無条件に実行される

```
(x < 10) -> y = 99;
```

上記のPromelaコードは以下のCのコードと等価

```
while (x >= 10); /* wait */
y = 99;
```

実行可能性

- チャネルからの受信はキューにデータがあれば実行可能、空ならばブロック
- チャネルに対する送信はキューに空きがあれば実行可能、一杯ならブロック

ランデブー

- chan c = [0] of {int};
- 長さ０のチャネルを使用する
- 送信側は受信されるまでブロックされる
- 受信側は送信されるまでブロックされる

assertによる検証

assert()

- モデルの適当な場所に挿入する事によって、その制御ポイントで、ある性質が成立しているか否かを検査する
- 括弧の中の式が偽ならば、エラーが発生する

SPINの状態管理

- グローバル状態ベクトルで管理される
- 全てのグローバル変数の値（メッセージチャネルを含む）
- 全てのプロセスの状態
  - 全てのローカル変数の値
  - プログラムカウンタの値

デモ

まとめ

モデル検査の効果

- 設計段階で並列処理の論理的・機能的欠陥を発見できる

副次的効果

- 並列処理において、共有資源を安全に操作することの難しさ、大切さがよくる。
- 設計が慎重になる

モデル作成のためのヒント

- 共有メモリのモデル化にはグローバル変数を使用する
- 遅れを持つような持通信路のモデル化にはチャネルを使用する

効率的なモデル作成のためのヒント

- プロセスの数を少なくする
- データオブジェクトの数を少なくする
- データオブジェクトのサイズを小さくする
  - intよりもbyte. byteよりもbit
- atomicな処理を多くする
- 但し、問題の本質を失わないように注意

- In a well-designed system, erroneous behavior should be impossible, not just improbable.

  Gerard J Holzmann,
  The SPIN MODEL CHECKER,
  p454

- 正しく設計されたシステムに於いては、誤りは起きてはならない。単に起きにくいというだけではなく。

ジェラルド・J・ホルツマン
SPINモデルチェッカー p454

情報の入手先

- http://www.spinroot.com
- 上記サイトより日本語に翻訳されたマニュアルも入手できる

参考図書

- 中島震著「SPINモデル検査」近代科学社

参考図書

- Holzmann, The Spin Model Checker: Primer and Reference Manual Addison-Wesley 2004, ISBN 0-321-22862-6, 608 pgs
- Ben-Ari, Principles of the Spin Model Checker, Springer, 216 pgs
- Model Checking, Clarke, Grumberg, Peled MIT Press, 1999

みんなでモデル検査を
やってみよう！