自己修復型人工物のための分散情報処理

Distributed information processing for self-restoring artifacts.

東京大学大学院　小松　謙介　Kensuke KOMATSU　Non-Member
東京大学　高橋　浩之　Hiroyuki TAKAHASHI　Member
東京大学　中沢　正治　Masaharu NAKAZAWA　Non-Member

Abstract　In the environment such as outer space where we cannot maintain the artifacts, it is the robustness that is important for artifacts. Thus the system which has the information processing circuit realizing self-maintenance function is desirable. So far some approach is tried to realize self-maintenance function using the qualitative reasoning technique. In this study, we try to apply distributed information processing technique to this qualitative reasoning information processor, by arranging many operating units with relatively low performance onto an application specific integrated circuit(ASIC). Now we are investigating the way mounting the self-maintenance logic by realizing the basic operating processor onto an FPGA(Field Programmable Gate Array) .

Keywords:　Distributed processing unit, Application Specific Integrated Circuit
E-mail: komatsu@sophie.q.t.u-tokyo.ac.jp

１．緒言

宇宙空間における人工物、例えば人工衛星はそのメンテナンスが非常に困難を極める。このような状況においては人工物のロバスト性が人工物の効用を大きく左右する重要な要素となってくる。

しかし原理的に故障を伴わない人工物は実現が不可能であるので、故障の発生を前提とした上で故障に耐性のある人工物という概念が提唱されるようになった。人工物が故障に耐性をもつためにはシステム内に自己修復機能をもつことが望ましく、定性推論の手法を用いて実現されてきた。これまでに行われている研究について、本研究では、この定性推論の情報処理部に対して、専用集積回路を用い、比較的低機能の演算器を物理モデルに即して多数配置し、分散情報処理を適用することを試みる。

２．自己修復型人工物ための情報処理

2.1 自己修復機能

本自己修復機能を実現するには、人工物の故障を検知しなければならない。情報処理回路は、人工物の挙動をモデル化し、人工物の各部分から得られた情報とそのモデルの間の関係を判断することによって故障を検知する。

2.2 定性推論

定性推論とは現象の定性的な側面に注目して推論を行っていくものである[1]。例えば空気に関して、$PV = nRT$　という式と現在の温度と圧力から、体積を厳密に求める方法を定量的推論とすると、定性推論では「Ｔが増えればＶも増える」「Ｔ↑⇒Ｖ↑」とより抽象化して推論を行う。自己修復機能に必要なのは故障部分を推定することであり、特定の部分に関する厳密な情報ではない、よって定性推論が有効に用いられてきた。

2.4 回路の構想

一般的な（ノイマン型の）計算機は処理装置及び主記憶装置から成り立つ。一方、定性推論型の演算を行うための装置について考えてみると、ここの演算それ自体は大まかなものでよく、それほど高い性能は求められない。すなわち定性推論演算においては演算の種類がある程度限定されるので、現在の標準的なＣＰＵの能力を生かしきれているとは言い難い。また定性推論演算においては扱う情報のサイズも定量的演算に比べてかなり小さくなるので現在の標準的なメモリでもオーバースペックである。し

かし、人工物が複雑になればなるほど物理モデルは複雑化し、その演算の量は極めて多くなり、既存のＣＰＵでは計算が追いつかなくなり、物理モデルの縮小化を目指したモジュール化などが行われている。以上をまとめると、従来の計算機は定性推論型の演算には最適ではないといえる。本研究では自己修復型人工物に適した、新しい分散型情報処理チップを実現することを目指して研究をすすめている。これは、定性推論のための演算に必要十分な程度の、比較的低機能で回路サイズの小さい定性演算機を多数用意し、独立に動作させることができるようにするもので、物理モデルに応じて、それらの定性演算機間の結線を行い、ネットワークを組み上げる、高速かつ大量の情報処理を実現するものである。

本研究で作成を目指す情報処理回路は、自己修復型人工物の活躍すべき極限環境下における動作を考慮し、情報処理回路それ自体の故障に対処することを考えた、以下のようなものである(Fig.1)。

まず回路の外延に並べられた比較回路が、外部から入力される定量的な情報を、「多い」「少ない」「正」「負」「ゼロ」等のより単純で抽象度の高い定性的な情報に変換する。

回路の中央部には定性推論のための簡単な演算をするセルが数多く並べられている。各セルは、個々がそれぞれ決められた簡単な演算を行う専用回路である。挙動のモデルから導かれる値と、外部から入力される定性値の間に矛盾がないか推論を行う。

セル間でもネットワークが形成されており、セル同士で演算結果に矛盾が生じた場合には多数決が行われたり、互いの信頼度が変化する機能も実装される。

本回路の利点は、リアルタイムでの並列情報処理が可能である点である。また回路上のあるセルが故障したとしても、故障を含むユニット以外には影響が生じない。

Table.1 に比較回路および定性演算用の回路例を ASIC(Application Specific Integrated Circuit)用のハードウェア記述言語である Verilog-HDL で記述したものを示す。このようなモジュールを組み合わせることにより大規模な演算処理が可能になると考えられる。

３．回路の試作

上記の回路を実現するための第一歩として、４９個のフォトトランジスタ（以後センサ）を並べ、各センサが隣接するセンサの信頼性を評価し、信頼性が低くなったから断線するモジュールを FPGA で作成した。開発環境は Quartus II、設計言語は Verilog-HDL である。

各センサは１ビットの物理情報しか持たないため、物理情報を定性値に変換する必要はない。隣接するセンサから得られる情報は同じはずなので隣接するセンサから得られる値を比較し、信頼性を決定する。

上記のことをふまえ以下の回路を作成した。

- 各センサは１６点の信頼度を持っている。
- 上下左右の４つセンサと１クロックごとに値を比べて値が異なるなら互いの信頼度を１つ下げ、同じなら１つ元にもどす。
- センサの信頼度が４点以下になると、まわりと断線する。

この回路を用い、フォトトランジスタ全体に光を当てながら、一部のセンサを壊した場合の影響を確かめた。結果は以下の通りであった。ほぼシミュレーション通りの結果となった。

- 故障したセンサはセンサ群から断絶された。
- 故障したセンサに正常なセンサが囲まれてしまうと正常なセンサが断線された。

また上と同じ回路を ASIC にするため設計ファイルの製作を行っている(Fig.2)。論理合成には DesignCompiler、配置配線には Apollo、レイアウトデータの出力には Milkyway を利用した。

４．結言

自己修復型人工物のための分散情報処理を実現する、定性演算器の仕様を考案した。また基本的なモジュールをハードウェア設計言語 Verilog-HDL で記述したものを FPGA や ASIC にした。今後はより複雑な物理モデルに対応した情報処理回路を設計し、その評価を行う予定である。

```
'define PLUS 3'b000
'define MINUS 3'b001
'define ZERO 3'b010
'define UNKNOWN 3'b011
'define UNDEF 3'b100

//定性値変換モジュール
module q_tran(ret,in)
 parameter WIDTH ;
 paremeter THRESHOLD ;

 output [2:0] ret;
 input [WIDTH-1:0] in ;
 reg [2:0] q;

 always @ (in)
  begin
   if(in>'THRESHOLD)
    begin
     q <= 'PLUS;
    end
   else if(in='THRESHOLD)
    begin
     q <= 'ZERO;
    end
   else if(in<'THRESHOLD)
    begin
     q <= 'MINUS;
    end
   else
    begin
     q <= 'UNKNOWN;
    end
  end
 assign ret = q;

endmodule

// 定性和モジュール
module q_plus(ret,in1,in2);
output [2:0] ret;
input [2:0] in1,in2;
reg [2:0] q;

 always @ (in1 or in2)
  begin
   casex({in1,in2}))
    {'UNKNOWN,2'bxx} : q<='UNKNOWN;
    {2'bxx,'UNKNOWN} : q<='UNKNOWN;
    {'ZERO,2'bxx} : q<='ZERO;
    {2'bxx,'ZERO} : q<='ZERO;
    {'PLUS,'MINUS} : q<='UNKNOWN;
    {'MINUS,'PLUS} : q<='UNKNOWN;
    {'PLUS,'PLUS}  : q<='PLUS;
    {'MINUS,'MINUS}: q<='MINUS;
    default       : q<='UNKNOWN;
   endcase
  end

 assign ret = q;

endmodule
```

Table 1　モジュール記述例

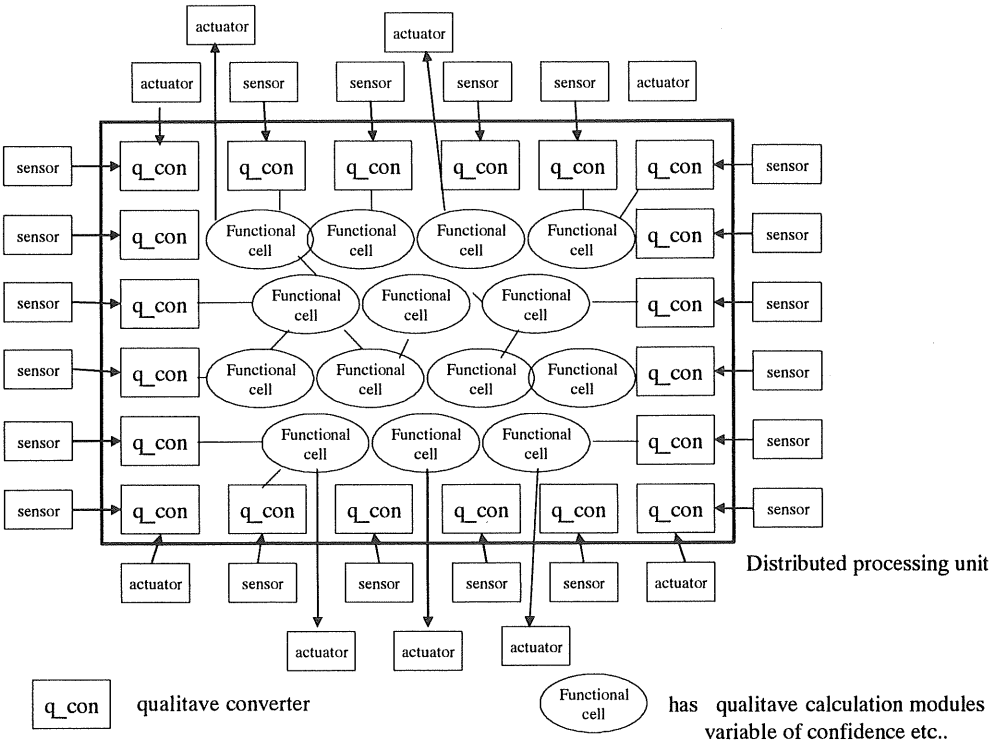

Fig. 1　分散型情報処理回路

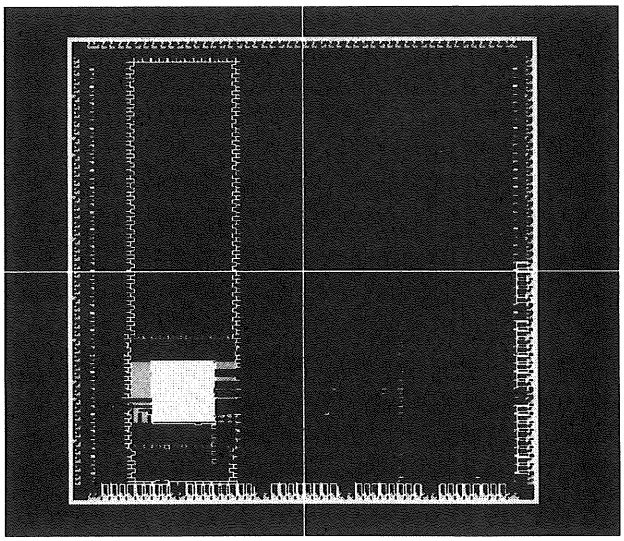

Fig. 2　ASIC レイアウトデータ

＊ASIC を設計は東京大学 VDEC を通し、シノプシス株式会社、ケイデンス株式会社の協力で行われました。

参考文献

[1] 西田豊明:「定性推論の諸相」,1993.