Automatic Synthesis of Branch Prediction Schemes through Genetic Programming

Cândea, C.
AI Research Group
Wittmann & Partner Comp. Sys.
2400 Sibiu, ROMANIA
ciprinac@airg.verena.ro

Staicu, M.
AI Research Group
Wittmann & Partner Comp. Sys.
2400 Sibiu, ROMANIA
mariuss@airg.verena.ro

Lucian N. Vinţan
University "L. Blaga",
Department of Computer Science
Emil Cioran 4 Sibiu, Romania
vintan@jupiter.sibiu.ro

Short Presentation

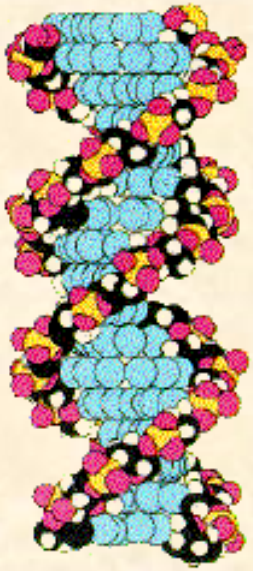
Started in 1998 like a small **research group**.
Now we are a **Research&Development company** with German capital.
Our **main products** are
Search Engine Agents (http://airg.verena.ro/sea)
Product Knowledge Management (http://www.wittmann-edv.de)
Also we are involved in some **research projects** like RoboCup simulation league.
(http://airg.verena.ro)

Overview

1. Motivation
2. Current trends
3. BPL Introduction
4. BPL Functions
5. Predictors samples
6. Genetic Algorithm
7. System Architecture
8. Implementation
9. Experimental Results
10. Conclusion and future work

1. Motivation

Predictors (scheme) made by human



Predictors (scheme) “Dolly”

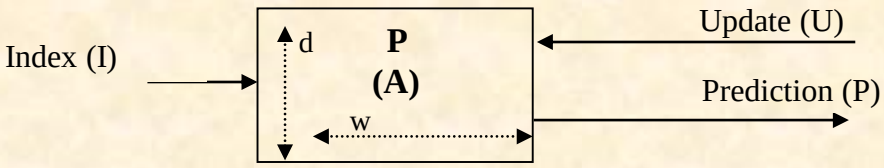
An integrated approach in Computer Architectures

2. Current trends

- ✓ Branch Target Buffer (BTB) structures
- ✓ Two Level Adaptive Branch Prediction
  - ➢ History Register (HR)
  - ➢ Pattern History Table (PHT)
- ✓ Neural branch prediction
- ✓ Markov chains predictors
- ✓ Automatic Synthesis of Branch Prediction Schemes

3. BPL Introduction

Memory Structure



Name	Description	Type
w	Width	Static
d	Depth	Static
i	Index for prediction and update	Dynamic
u	Update value	Dynamic

Language Terminals

Terminal Name	Description
T	Taken/Not Taken value read from trace file (1 bit value)
PC	Current program counter
A	Refer first A type memory definition
P	Refer first P type memory definition
0	0
1	1

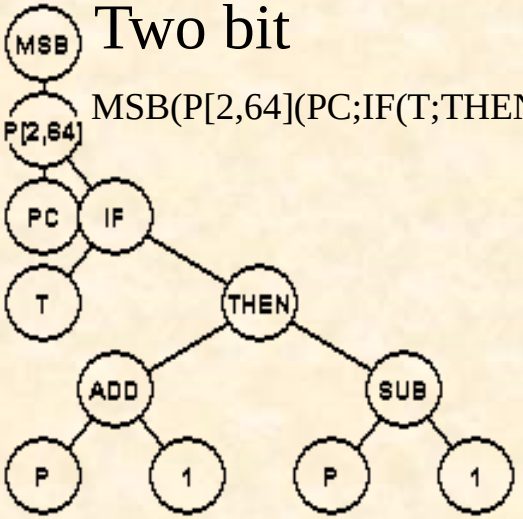
4. BPL Functions

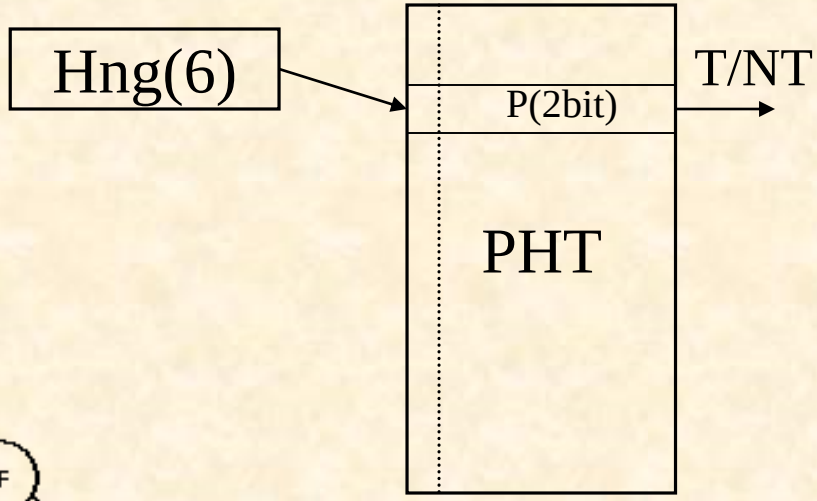
Functions	Description
MSB	Most significant bits
CAT	Concatenation
XOR	Xor function
MASKHI	Most n significant bits
MASKLO	Most less n significant bits
SUB	Saturating subtraction (0 val. min)
ADD	Saturation adding
EQU	Equality of two values
IF	If
THEN	Then
PLUS	Adding
MINUS	Subtraction

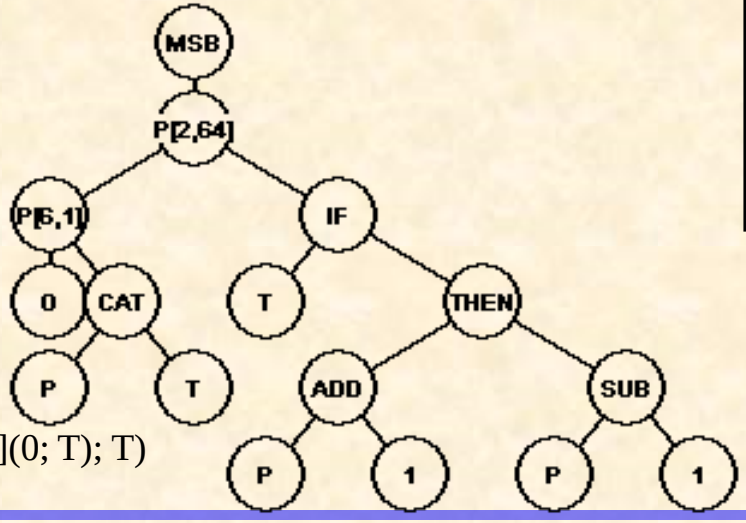
5. Predictors samples

Two bit

MSB(P[2,64](PC;IF(T;THEN(ADD(P;1);SUB(P;1)))))







GAg

Gag[n](; T) = Twobit[2^n](Hist[n,1](0; T); T)

6. Genetic Algorithm

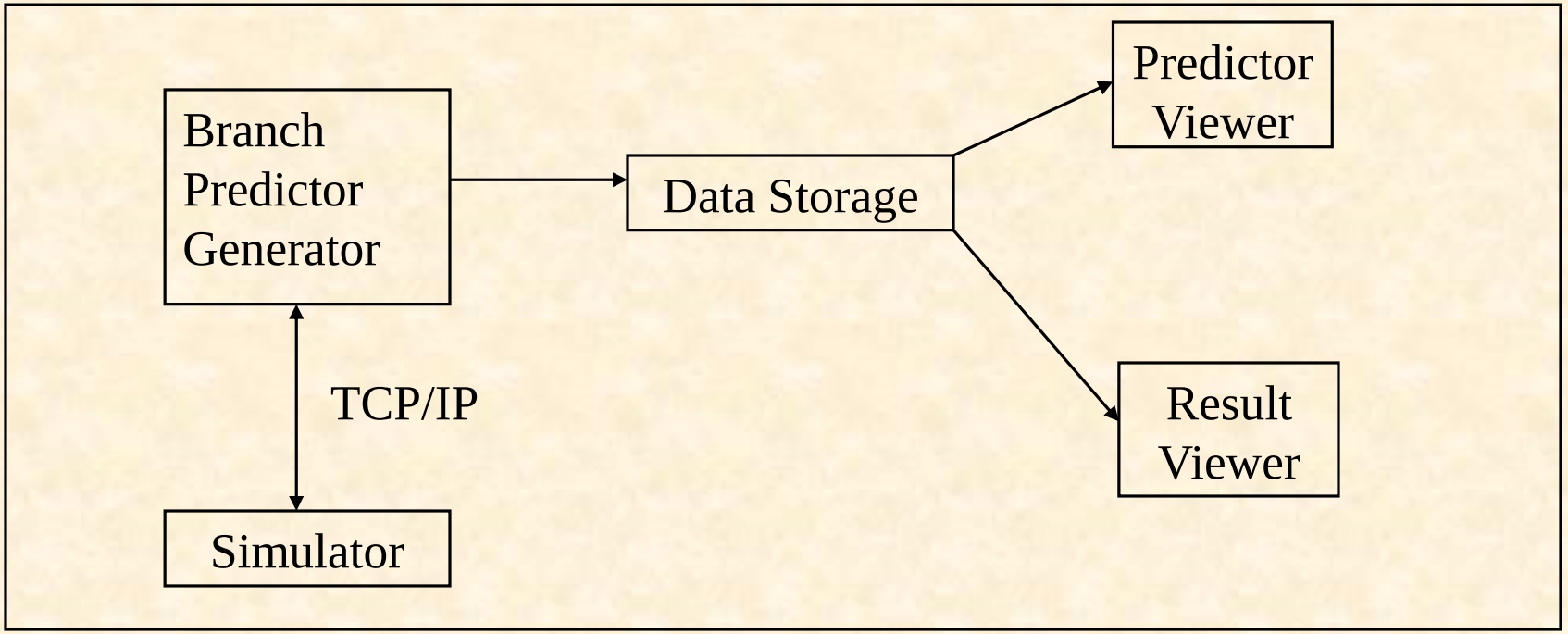
Algorithm steps

- 1. Create initial population of randomly generated individuals
- 2. Rank fitness of individuals in the population by simulation
- 3. Apply genetic operations to create new generation
- 4. If no stop condition (e.g. if user send a stop command) go to step 2

Genetic operators

- Integrity check
- Replication
- Crossover
- Mutation

7. System architecture



General System Structure

8. Implementation

Cluster used for simulations

5 Computers PIII 500Mhz
196 MB RAM
5Gb HDD
Windows 2000

- 1 Server
- 4 Clients

- Simulate for maximum 5 generation
- Each population has 400 chromosomes
- Simulation time for 1 generation is 60 seconds
  - 60 seconds * 400 chromosomes => 400 min ~ 6.6 hours * 5 generation ~ 33.3 hours ~ 1.4 days

9. Experimental results

Predictor	Rate
Onebit[1,2K]	0.664
Twobit[2,64K]	0.746
Gag[2]	0.849
Pag[18,8K]	0.825
Pap[9,18,8K]	0.771

Predictor	Rate
MSB(MASKHI(ADD(A[23,64](0;T);A[1,128](PC;PC));0))	0.711
MSB(EQU(MSB(MASKHI(ADD(IF(XOR(0;1);THEN(PC;PC));SUB(MSB(A[22,64](1;1));MASKLO(0;EQU(PC;PC))));PC));0))	0.759
MSB(EQU(XOR(SUB(MINUS(0;1);IF(ADD(A[31,64](A;PC);1);THEN(1;PC)));EQU(PC;PC));ADD(A[24,32](0;1);MASKHI(MSB(1);PC))))	0.811

For simulation we use the Stanford traces

10. Conclusions and future work

- Can describe a variety of predictors, also manipulate them very easy
- Predictor obtained automatically are comparable with the “hand made”
- Are too complex
- Some interesting new ideas useful in human designed branch predictors

- Extend the language with other components
- Adapt current version in order to find indirect jumps predictors
- Adding other genetic operators
- Algorithm for reduce the branch complexity

http://airg.verena.ro