Evolutionary Search in Machine Learning

Lutz Hamel

Dept. of Computer Science & Statistics

University of Rhode Island

What is Machine Learning?

- Programs that get better with *experience* given some *task* and some *performance measure.*
- Most common is *inductive learning,* that is learning from a set of positive and negative examples or facts.
- Learning to:
  - classify customers;
  - recognize spoken words;
  - play games.

Machine Learning Today

- Today's machine learning tools are "single-table" oriented:
  - attribute-value oriented
  - objects are represented by a *fixed set* of attributes.
- Here we consider learning using first-order logic as representation, instead of just attribute values;
  - propositional vs. predicated representation

First-Order Equational Logic

Equational logic is the logic of substituting equals for equals with algebras as models and term rewriting as the operational semantics.

```
theory LIST is
  sort List .
  protecting INT .

  op cons : Int List -> List .
  op nil  : List .
  op length : List -> Int .

  var I : Int .
  var L : List .

  eq length(nil) = 0 .
  eq length(cons(I,L)) = 1 + length(L) .
end
```

<u>A Deduction:</u>

length(cons(3,cons(2,nil)))

{equation 2: I $\leftarrow$ 3, L $\leftarrow$ cons(2,nil)}

$\Rightarrow$ 1 + length(cons(2,nil))

{equation 2: I $\leftarrow$ 2, L $\leftarrow$ nil}

$\Rightarrow$ 1 + 1 + length(nil)

{equation 1}

$\Rightarrow$ 1 + 1 + 0

{INT module: basic arithmetic}

$\Rightarrow$ 2

Inductive Equational Logic

- In inductive equational logic we induce equational theories (hypotheses) from equations which represent the facts.
- This seems to be opposite of what we do in ordinary (deductive) logic -deduce facts from theories.

$$\text{Theories} \quad \underset{\text{Induction}}{\overset{\text{Deduction}}{\rightleftarrows}} \quad \text{Facts}$$

Equational Induction

Given a fact theory $F = P \cup \neg N$, where

P represents the positive examples,

N represents the negative examples,

and B represents background or domain information,

then a Hypothesis is a theory H which explains all the facts using the background theory B, formally:

$$H \cup B \vdash f, \forall f \in F$$

Example: The Predicate Even

```
theory EVEN-FACTS is
  sort Int .
  op 0 : -> Int .
  op s : Int -> Int .
  op even : Int -> Bool .

  eq even(0) = true .
  eq even(s(s(0))) = true .
  eq even(s(s(s(s(0))))) = true .
  eq even(s(0)) = false .
  eq even(s(s(s(0)))) = false .
  eq even(s(s(s(s(s(0)))))) = false .
end
```

```
theory EVEN is
  sort Int .
  op 0 : -> Int .
  op s : Int -> Int .
  op even : Int -> Bool .
  var X : Int .

  eq even(s(s(X))) = even(X) .
  eq even(0) = true .
end
```

Implementation: Genetic Programming

Implemented in the OBJ3 System:

- Compute an initial (random) population of candidate theories.
- Evaluate each candidate theory's fitness using the OBJ3 rewrite engine:

  $$fitness(T) = facts^2(T) + 1/length(T)$$

- Perform candidate theory reproduction according to the genetic programming paradigm: *crossover* & *mutation*.
- Compute new population of candidate theories.
- Goto Step 2 or stop if target criteria have been met.

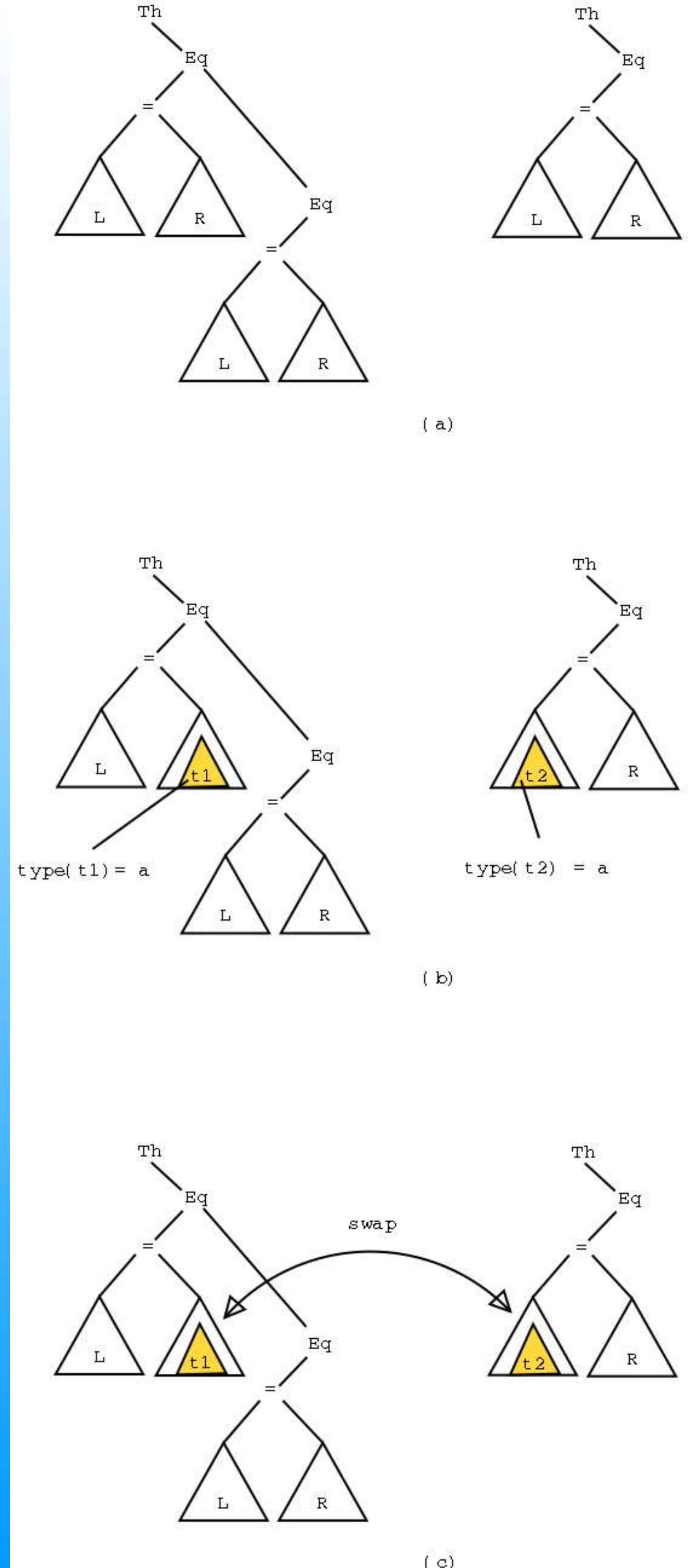
Crossover

Crossover -breed a new theory based on the structure of two parent theories.

Theories are represented as abstract syntax trees in memory.

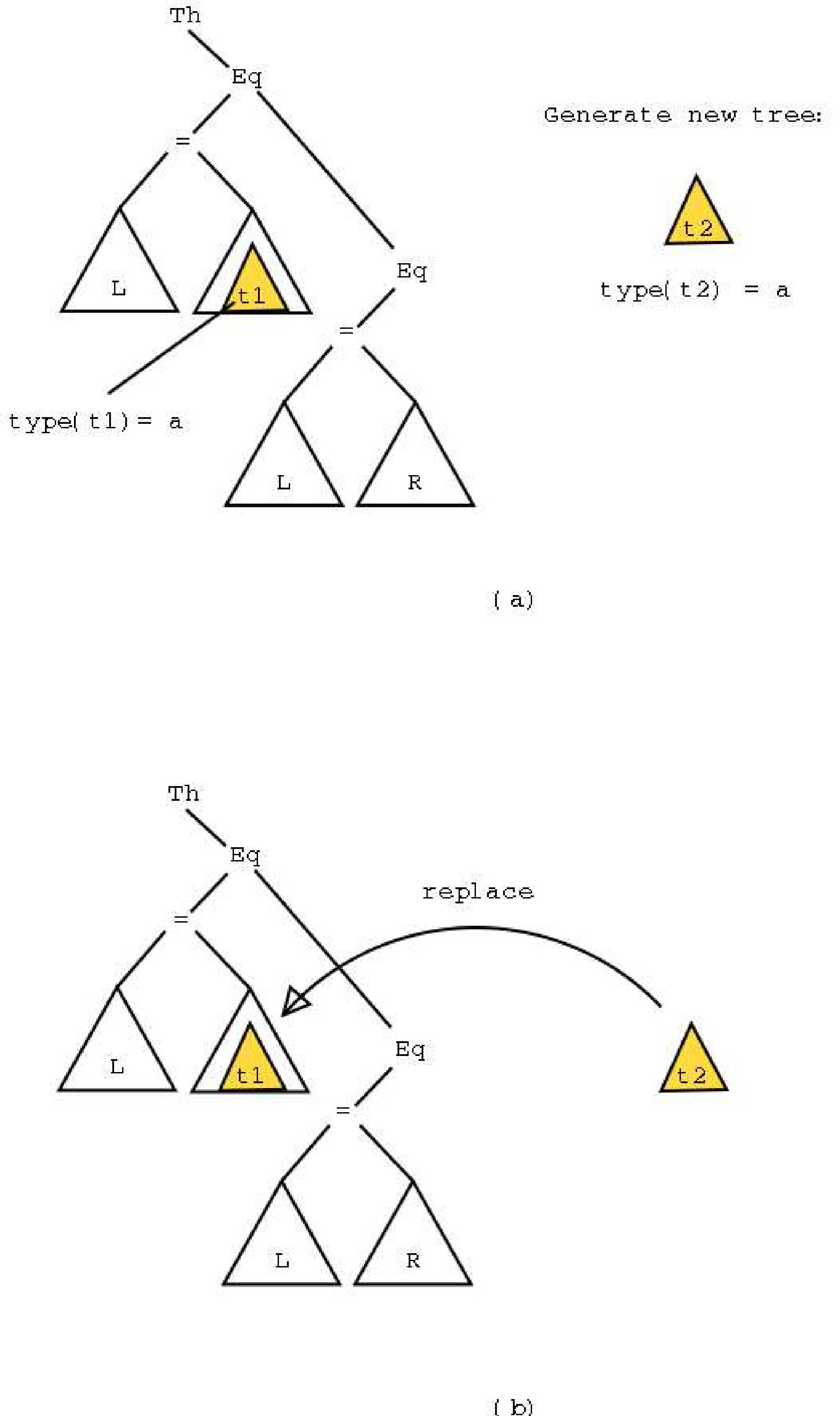
The amount of crossover and mutation is governed by preset parameters.

Strongly typed equational logic -therefore we need to be careful with the types and generate appropriate subtrees.



Mutation

Mutation -breed a new theory based on a single parent with a single mutation in the abstract syntax tree.



Experiment I: Multi-Objective Learning

```
theory STACK-FACTS is
  sorts Stack Element .
  ops a b c d: -> Element .
  op v : -> Stack .
  op top : Stack -> Element .
  op pop : Stack -> Stack .
  op push : Stack Element -> Stack .

  eq top(push(v,a)) = a .
  eq top(push(push(v,a),b)) = b .
  eq top(push(push(v,b),a)) = a .
  eq top(push(push(v,d),c)) = c .

  eq pop(push(v,a))= v .
  eq pop(push(push(v,a),b)) = push(v,a) .
  eq pop(push(push(v,b),a)) = push(v,b) .
  eq pop(push(push(v,d),c)) = push(v,d) .
end
```

Comparison:
The FLIP system did not produce a solution at all!

Statistics:
- Evolution over 50 generations.
- Population of 150 individuals.
- Converged to canonical stack theory in 20 of 150 runs.
- Convergence rate ~15%.

```
theory STACK is
  sorts Stack Element .
  ops a b c d: -> Element .
  op v : -> Stack .
  op top : Stack -> Element .
  op pop : Stack -> Stack .
  op push : Stack Element -> Stack .
  var S : Stack .
  var E : Element .

  eq top(push(S,E)) = E .
  eq pop(push(S,E)) = S .
```

Experiment II: Learning in Noise

```
theory EVEN-FACTS is
  sort Int .
  op 0 : -> Int .
  op s : Int -> Int .
  op even : Int -> Bool .
  eq even(0) = true .
  eq even(s(s(0))) = true .
  eq even(s(s(s(s(0))))) = true .
  eq even(s(0)) = false .
  eq even(s(s(0))) = false .
  eq even(s(s(s(0)))) = false .
  eq even(s(s(s(s(s(0)))))) = false .
end
```

Definition:
Noise are inconsistencies in a fact theory.

Statistics:

- Evolution over 50 generations.
- Population of 150 individuals.
- Converged to canonical theory in 41 of 50 runs.
- Convergence rate ~80%.

```
theory EVEN is
  sort Int .
  op 0 : -> Int .
  op s : Int -> Int .
  op even : Int -> Bool .
  var X : Int .
  eq even(s(s(X))) = even(X) .
  eq even(0) = true .
end
```

Comparison:
The FLIP system produced an incorrect solution!

Summary

- Here we presented a framework for machine learning with logic.
- The logic we considered was equational logic -*inductive equational logic*
- We sketched a prototype implementation based on genetic programming.
- We showed that this implementation seems to be very robust in multi-objective learning and learning in the presence of noise.
- This work will appear in the proceedings of the CEC2003 conference.