Reliable and Interpretable Artificial Intelligence
Exercise 2

Dana Drachsler-Cohen
ETH Zurich

Fall 2017
Exact Programming by Example

• Programming by examples is the task of synthesizing a program from a set of input-output examples
  • The hope is that the examples are representative enough for the heuristic to synthesize the correct program

• Exact programming by examples is the two-step task:
  1. Learning a mathematical formula capturing the user intent on all inputs
  2. Synthesizing a program meeting the learned specification

• The learning step is phrased in the setting of exact learning
  • Reason about the number of queries compared to the optimum/lower bound
We defined a hypothesis space in which nodes are representative of the equivalence classes and edges link implied nodes.

\[ \text{isMale} \lor \text{isFemale} \lor \text{isAdult} \lor \text{isMan} \lor \text{isWoman} \lor \text{isChild} \lor \text{isBoy} \lor \text{isGirl} \]
The D-SPEX Algorithm

• We defined the notion of a witness, based on which showed D-SPEX
Complexity Analysis and Lower Bound

Theorem: If the children and witnesses of any $\varphi \in N$ can be found in time $t$, then D-SPEX learns $\psi$ in time $t \cdot |Q|$ and $|Q| \cdot \max_{n \in N}(\text{children}(n))$ membership queries.

Theorem: any learning algorithm that learns $Q_V$ must pose at least $\max(\log(|N|), \max_{n \in N}|\text{Children}(n)|)$. In particular, D-SPEX poses at most $|Q| \cdot \text{OPT}(Q_V)$. 
Today: An End-to-End Exact PBE

**Technical Analysis:** Predict price direction using current prices

**Patterns:** Special forms that signal whether to buy or sell

**Goal:** Synthesize a program (a model) detecting a pattern

\[
P_0 = \text{LLV}(\text{Close}, W); \\
P_1 = \text{HHV}(\text{Close}, BP_0); \\
P_2 = \text{LLV}(\text{Close}, BP_1); \\
P_3 = \text{HHV}(\text{Close}, BP_2); \\
P_4 = \text{LLV}(\text{Close}, BP_3); \\
P_5 = \text{HHV}(\text{Close}, BP_4); \\
P_6 = \text{LLV}(\text{Close}, BP_5); \\
\text{Filter} = P_0 < P_1 \land P_2 < P_1 \land P_1 < P_3 \land P_5 < P_3 \land P_4 < P_5 \land P_6 < P_5;
\]

**Idea:** Learn the exact pattern from charts

**Challenges:**
Which questions to ask? How to reduce their number?

**Solution:**
Algorithm that will ask at most \(|F| \cdot \text{OPT}(F_v)\) membership queries where \(F\) is the set of possible features, \(\text{OPT}(F_v)\) is the minimum worst case number of membership queries for \(F_v\).
Time-Series Patterns from Charts

Goal: an exact PBE that learns patterns in time-series charts

Time-series charts are used in many domains including financial analysis, medicine, and seismology.
Time-Series Patterns from Charts

Experts use these charts to predict important events (e.g., trend changes in a stock price) indicated by special patterns.
Time-Series Patterns from Charts

There is a lot of study on common patterns and there are many softwares that enable these experts to write a program that alerts upon detecting their customized pattern.

- **Head and Shoulders**: A bearish reversal pattern consisting of a peak (the head) and two lower peaks (the shoulders).
- **Cup and Handle**: A bullish reversal pattern where the price forms a cup shape followed by a handle that breaks above the cup's top.
Time-Series Patterns from Charts

Unfortunately, writing programs is a complex task for these experts, who are not programmers

Goal: learn the specifications from chart examples, then synthesize a program

\[
\begin{align*}
\text{P}_0 &= \text{LLV}(\text{Close}, W); \\
\text{BP}_0 &= \text{LLVBars}(\text{Close}, W); \\
\text{P}_1 &= \text{HHV}(\text{Close}, \text{BP}_0); \\
\text{BP}_1 &= \text{HHVBars}(\text{Close}, \text{BP}_0); \\
\text{P}_2 &= \text{LLV}(\text{Close}, \text{BP}_1); \\
\text{BP}_2 &= \text{LLVBars}(\text{Close}, \text{BP}_1); \\
\text{P}_3 &= \text{HHV}(\text{Close}, \text{BP}_2); \\
\text{BP}_3 &= \text{HHVBars}(\text{Close}, \text{BP}_2); \\
\text{P}_4 &= \text{LLV}(\text{Close}, \text{BP}_3); \\
\text{BP}_4 &= \text{LLVBars}(\text{Close}, \text{BP}_3); \\
\text{P}_5 &= \text{HHV}(\text{Close}, \text{BP}_4); \\
\text{BP}_5 &= \text{HHVBars}(\text{Close}, \text{BP}_4); \\
\text{P}_6 &= \text{LLV}(\text{Close}, \text{BP}_5); \\
\text{Filter} &= \text{P}_0 < \text{P}_1 \text{ AND } \text{P}_2 < \text{P}_1 \text{ AND } \text{P}_1 < \text{P}_3 \text{ AND } \text{P}_3 < \text{P}_5 \text{ AND } \text{P}_5 < \text{P}_5 \text{ AND } \text{P}_6 < \text{P}_5;
\end{align*}
\]
Time-series Patterns

A chart is a function over time
\[ p_i \] is the price at time point \( i \)

A pattern is a conjunction over \( Q = \{ p_i < p_j | i \neq j \} \)

\[ \varphi_{HS} = (p_0 < p_2) \land (p_2 < p_1) \land (p_1 < p_3) \land (p_2 < p_4) \land (p_4 < p_5) \land (p_6 < p_5) \land (p_5 < p_3) \land (p_6 < p_0) \]
Exact PBE for Time-series Patterns

• We will assume a slightly different setting
• The learning begins from the user who provides an initial chart example $e$
• Then, the set of predicates is $Q_e = \{p_i < p_j | e \models p_i < p_j\}$
• Target formula is in $Q_{e,\wedge} = \{\wedge_{q \in P} q | P \subseteq Q_e\}$
• Note that $Q_{e,\wedge}$ cannot contain cyclic constraints
  • $p_i < p_j \in Q_{e,\wedge} \Rightarrow p_j < p_i \notin Q_{e,\wedge}$
HW: Exact PBE for Time-series Patterns

1. Define C-SPEX, a variation of D-SPEX for learning conjunctions
   • Hint: What is the hypothesis space? How do the lemma change?

2. Let $e$ be a chart example and $Q_e = \{ p_i < p_j \mid e \models p_i < p_j \}$
   a) Define how to compute the children of a node. What is the time complexity?
   b) Define how to find the witnesses. What is the time complexity?
      • Hint: topological sorting
   c) Determine how many membership queries C-SPEX can present.
      • Hint: we have a much better bound...
The Hypothesis Space

• $N = \{ \land_{q \in A} q | \forall q' \in Q_e \setminus A: (\land_{q \in A} q \land q') \not\equiv \land_{q \in A} q \}$
• $E = \{ (\varphi_1, \varphi_2) | \varphi_1 \models \varphi_2 \land \forall \varphi_3 (\varphi_1 \models \varphi_3 \land \varphi_3 \not\equiv \varphi_2 \rightarrow \varphi_3 \not\models \varphi_2) \}$
• Lemma 1: If $\varphi_1$ is a descendant of $\varphi_2$, then $Q(\varphi_1) \subset Q(\varphi_2)$
• Lemma 2: $Q(\gcd(\varphi_1, \varphi_2)) = Q(\varphi_1) \cap Q(\varphi_2)$
  • In particular, $\land (Q(\varphi_1) \cap Q(\varphi_2)) \in N$
Witnesses

• A witness for \( \varphi_1, \varphi_2 \in N \) is an example \( e \in D \) such that \( \varphi_1(e) \neq \varphi_2(e) \)

• Lemma 3: Let \( \varphi_1 \) be a child of \( \varphi_2 \) and \( e \) a witness for them. Then:
  1. \( \varphi_1(e) = 1, \varphi_2(e) = 0 \)
  2. For every \( q \in Q(\varphi_1), q(e) = 1 \)
  3. For every \( q \in Q(\varphi_2) \setminus Q(\varphi_1), q(e) = 0 \)

• Lemma 4: Let \( \{\varphi_1, \ldots, \varphi_k\} \) be the children of \( \varphi \). If \( e_i \) is a witness for \( \varphi_i \) and \( \varphi \), then \( e_i \) is not a witness for \( \varphi, \varphi_j \) for any \( j \neq i \).

• Lemma 5: Let \( \varphi_i \) be a child of \( \varphi \), \( e_i \) a witness, \( \varphi_j \) a descendant of \( \varphi \)
  1. If \( \varphi_j(e_i) = 1, \varphi_j \) is a descendant of \( \varphi_i \) or equal to \( \varphi_i \)
  2. If \( \varphi_j(e_i) = 0, \varphi_j \) is not a descendant of \( \varphi_i \) nor equal to \( \varphi_i \)
The C-SPEX Algorithm

\textbf{C-SPEX(}\varphi,\mathcal{T}):\quad //\text{ initially, call with}\text{ C-SPEX}(\land_{q\in\mathcal{Q}} q, \emptyset)

\begin{align*}
Q & \leftarrow Q(\varphi); \\
\text{Flag} & \leftarrow 1 \\
\text{For} \ \varphi' \in \text{children}(\varphi): \\
\quad \text{If} \ (\forall R \in \mathcal{T}. Q(\varphi') \not\subseteq R) \\
\quad \quad e & \leftarrow \text{witness}(\varphi, \varphi') \\
\quad \quad \text{If} \ (\text{mem}(e) = 1) \\
\quad \quad \quad Q & = Q \cap Q(\varphi'); \text{Flag} \leftarrow 0 \\
\quad \quad \text{Else} \ T & \leftarrow T \cup \{Q(\varphi')\} \\
\quad \text{If} \ (\text{Flag} = 1) \text{ return } \land_{q\in\mathcal{Q}} q \\
\text{C-SPEX}(\land_{q\in\mathcal{Q}} q, \mathcal{T})
\end{align*}
HW: Exact PBE for Time-series Patterns

1. Define C-SPEX, a variation of D-SPEX for learning conjunctions
   • Hint: What is the hypothesis space? How do the lemma change?

2. Let $e$ be a chart example and $Q_e = \{p_i < p_j | e \models p_i < p_j\}$
   a) Define how to compute the children of a node. What is the time complexity?
      • Hint: represent the constraints in a graph whose nodes are $p_i$
   b) Define how to find the witnesses. What is the time complexity?
      • Hint: topological sorting
   c) Determine how many membership queries C-SPEX can present.
      • Hint: we have a much better bound...
Representing Nodes as Constraint Graphs

Given a node $n$, we define a graph $G_n$ capturing $n$’s constraints: nodes are points, there is an edge $(i, j)$ for every $p_i < p_j \in n$

\[
(p_0 < p_2) \land (p_2 < p_1) \land (p_1 < p_3) \land (p_2 < p_4) \land (p_4 < p_5) \land (p_6 < p_5) \land (p_5 < p_3) \land (p_6 < p_0)
\]
Computing the Node’s Children

The children of $n$ are not equivalent to $n$
That is, they have (at least) one less constraint... That is, one less edge in the graph
Can there be two fewer constraints?
What kind of constraints can be removed to obtain a child?

Lemma: Let $n$ be a node. For every $(i, j)$ in the graph $G_n$, such that $(i, j)$ are reachable only through this edge, $n \setminus \{p_i < p_j\}$ is a child of $n$
Computing a Node’s Children

Lemma: Let \( n \) be a node. For every \((i, j)\) in the graph \( G_n \), such that \((i, j)\) are reachable only through this edge, \( n \setminus \{p_i < p_j\} \) is a child of \( n \)

• To find the children, check for every edge \((i, j)\):
  • Is \( j \) reachable from \( i \) in the graph that does not contain \((i, j)\)
    • E.g., BFS
  • Time complexity \( O(E \cdot (E + N)) \)
HW: Exact PBE for Time-series Patterns

1. Define C-SPEX, a variation of D-SPEX for learning conjunctions
   • Hint: What is the hypothesis space? How do the lemma change?

2. Let $e$ be a chart example and $Q_e = \{p_i < p_j \mid e \models p_i < p_j\}$
   a) Define how to compute the children of a node. What is the time complexity?
      • Hint: represent the constraints in a graph whose nodes are $p_i$
   b) Define how to find the witnesses. What is the time complexity?
      • Hint: topological sorting
   c) Determine how many membership queries C-SPEX can present.
      • Hint: we have a much better bound...
Finding a Witness

• When learning conjunctive formulas, a witness is an example satisfying the child but not the parent
Finding a Witness

Let $n$ be a node and $n_{i,j}$ be a child of $n$.
In the graph $G_{n_{i,j}}$, there is no edge between $i, j$
Match $i, j$ to a single node, and get a DAG
Use topological sorting to find an assignment to the points
⇒This assignment satisfies $n_{i,j}$ but not $n$
1. Define C-SPEX, a variation of D-SPEX for learning conjunctions
   • Hint: What is the hypothesis space? How do the lemma change?

2. Let $e$ be a chart example and $Q_e = \{ p_i < p_j \mid e \models p_i < p_j \}$
   a) Define how to compute the children of a node. What is the time complexity?
      • Hint: represent the constraints in a graph whose nodes are $p_i$
   b) Define how to find the witnesses. What is the time complexity?
      • Hint: topological sorting
   c) Determine how many membership queries C-SPEX can present.
      • Hint: we have a much better bound...
How many times can a constraint be asked about?
If the classification is positive?
\[ p_i < p_j \] is not in the target formula, and is thus removed
If the classification is negative?
\[ p_i < p_j \] is in the target formula, and thus C-SPEX proceeds to its descendants, all include this constraint

**Lemma:** Given an initial example \( e \), the class \( Q_{e,\Lambda} \) is learnable in polynomial time and with at most \( |Q_{e,\Lambda}| \) membership queries
An End-to-End Exact PBE

• How can we obtain the end-to-end synthesizer?

P₀ = LLV(Close, W);
BP₀ = LLVBars(Close, W);
P₁ = HHV(Close, BP₀);
BP₁ = HHVBars(Close, BP₀);
P₂ = LLV(Close, BP₁);
BP₂ = LLVBars(Close, BP₁);
P₃ = HHV(Close, BP₂);
BP₃ = HHVBars(Close, BP₂);
P₄ = LLV(Close, BP₃);
BP₄ = LLVBars(Close, BP₃);
P₅ = HHV(Close, BP₄);
BP₅ = HHVBars(Close, BP₄);
P₆ = LLV(Close, BP₅);
Filter = P₀ < P₁ AND P₂ < P₁ AND P₁ < P₃ AND P₅ < P₃ AND P₄ < P₅ AND P₆ < P₅;
An End-to-End Exact PBE

From the specification infer the minimum & maximum points, and generate the code

The witnesses are assignments ($p_0, ..., p_n$) —display as chart

**Exact Learning**

- **$e \in \hat{H}$?**
- **yes/no**

**Synthesis from Specification**

- **Spec**
- **Function**

$P_0 = \text{LLV}(\text{Close}, W)$;
$BP_0 = \text{LLVBars}(\text{Close}, W)$;
$P_1 = \text{HHV}(\text{Close}, BP_0)$;
$BP_1 = \text{HHVBars}(\text{Close}, BP_0)$;
$P_2 = \text{LLV}(\text{Close}, BP_1)$;
$BP_2 = \text{LLVBars}(\text{Close}, BP_1)$;
$P_3 = \text{HHV}(\text{Close}, BP_2)$;
$BP_3 = \text{HHVBars}(\text{Close}, BP_2)$;
$P_4 = \text{LLV}(\text{Close}, BP_3)$;
$BP_4 = \text{LLVBars}(\text{Close}, BP_3)$;
$P_5 = \text{HHV}(\text{Close}, BP_4)$;
$BP_5 = \text{HHVBars}(\text{Close}, BP_4)$;
$P_6 = \text{LLV}(\text{Close}, BP_5)$;
$BP_6 = \text{LLVBars}(\text{Close}, BP_5)$;

Filter = $P_0 < P_1 \land P_2 < P_1 \land P_1 < P_3 \land P_5 < P_3 \land P_4 < P_5 \land P_6 < P_5$;

The witnesses are assignments ($p_0, ..., p_n$) —display as chart

$\left( p_0 < p_2 \right) \land \left( p_2 < p_1 \right) \land \left( p_1 < p_3 \right) \land \left( p_2 < p_4 \right)$
$\land \left( p_4 < p_5 \right) \land \left( p_5 < p_6 \right) \land \left( p_5 < p_3 \right) \land \left( p_6 < p_0 \right)$

From the specification infer the minimum & maximum points, and generate the code

**Exact Learning**

- **$e \in \hat{H}$?**
- **yes/no**

**Synthesis from Specification**

- **Spec**
- **Function**

$P_0 = \text{LLV}(\text{Close}, W)$;
$BP_0 = \text{LLVBars}(\text{Close}, W)$;
$P_1 = \text{HHV}(\text{Close}, BP_0)$;
$BP_1 = \text{HHVBars}(\text{Close}, BP_0)$;
$P_2 = \text{LLV}(\text{Close}, BP_1)$;
$BP_2 = \text{LLVBars}(\text{Close}, BP_1)$;
$P_3 = \text{HHV}(\text{Close}, BP_2)$;
$BP_3 = \text{HHVBars}(\text{Close}, BP_2)$;
$P_4 = \text{LLV}(\text{Close}, BP_3)$;
$BP_4 = \text{LLVBars}(\text{Close}, BP_3)$;
$P_5 = \text{HHV}(\text{Close}, BP_4)$;
$BP_5 = \text{HHVBars}(\text{Close}, BP_4)$;
$P_6 = \text{LLV}(\text{Close}, BP_5)$;
$BP_6 = \text{LLVBars}(\text{Close}, BP_5)$;

Filter = $P_0 < P_1 \land P_2 < P_1 \land P_1 < P_3 \land P_5 < P_3 \land P_4 < P_5 \land P_6 < P_5$;

The witnesses are assignments ($p_0, ..., p_n$) —display as chart

$\left( p_0 < p_2 \right) \land \left( p_2 < p_1 \right) \land \left( p_1 < p_3 \right) \land \left( p_2 < p_4 \right)$
$\land \left( p_4 < p_5 \right) \land \left( p_5 < p_6 \right) \land \left( p_5 < p_3 \right) \land \left( p_6 < p_0 \right)$

From the specification infer the minimum & maximum points, and generate the code