Declarative Merging of and Reasoning about Decision Diagrams

Thomas Eiter      Thomas Krennwallner      Christoph Redl

{eiter,tkren,redl}@kr.tuwien.ac.at

September 12, 2011
Outline

1 Motivation

2 Preliminaries: MELD

3 Merging of Decision Diagrams

4 Reasoning about Decision Diagrams

5 Application: DNA Classification

6 Conclusion
Outline

1 Motivation

2 Preliminaries: MELD

3 Merging of Decision Diagrams

4 Reasoning about Decision Diagrams

5 Application: DNA Classification

6 Conclusion
Motivation

Decision Diagrams

- Important means for decision making
- Intuitively understandable
- Not only for knowledge engineers

Examples

- Severity ratings (e.g. TNM system)
- Diagnosis of personality disorders
- DNA classification
Motivation

Multiple Diagrams

Reasons

- Different opinions
- Randomized machine-learning algorithms
- Statistical impreciseness

**Question:** How to combine them?
Multiple Diagram Integration

The DDM System

- Integration process declaratively described

- Ingredients:
  - 1 input decision diagrams
  - 2 merging algorithms
    (predefined or user-defined)

- Focus:
  - process formalization
  - experimenting with different (combinations of) merging algorithms
  - declarative reasoning for controlling the merging process

- We do **not** focus:
  - concrete merging strategies
  - accuracy improvement
Outline

1. Motivation
2. Preliminaries: MELD
3. Merging of Decision Diagrams
4. Reasoning about Decision Diagrams
5. Application: DNA Classification
6. Conclusion
MELD

Task

- Collection of knowledge bases: $KB = KB_1, \ldots, KB_n$
- Associated collections of belief sets: $BS(KB_1), \ldots, BS(KB_n) \in \mathcal{B}_\Sigma$
- Goal: Integrate them into a single set of belief sets

Method: Merging Operators

$$\circ^{n,m} : \left(\mathcal{2}^{\mathcal{B}_\Sigma}\right)^n \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_m \to \mathcal{2}^{\mathcal{B}_\Sigma}$$

Example

Operator definition:

$$\circ^{2,0}_{\cup} (\mathcal{B}_1, \mathcal{B}_2) = \{ B_1 \cup B_2 \mid B_1 \in \mathcal{B}_1, B_2 \in \mathcal{B}_2, \#A : \{ A, \neg A \} \subseteq (B_1 \cup B_2) \} ,$$

Application:

- $\mathcal{B}_1 = \{ \{ a, b, c \}, \{ \neg a, c \} \}$, $\mathcal{B}_2 = \{ \{ \neg a, d \}, \{ c, d \} \}$
- $\circ^{2,0}_{\cup} (\mathcal{B}_1, \mathcal{B}_2) = \{ \{ a, b, c, d \}, \{ \neg a, c, d \} \}$
MELD

Merging Plan

- Hierarchical arrangement of merging operators

Example

Diagram showing the hierarchical arrangement of merging operators with sets BS(KB1), BS(KB2), BS(KB3), BS(KB4), and BS(KB5).
### MELD

#### Merging Tasks

- User provides
  - belief bases with associated collections of belief sets
  - merging plan
  - optional: user-defined merging operators
- MELD: automated evaluation

#### Advantages

- Reuse of operators
- Quick restructuring of merging plan
Outline

1. Motivation
2. Preliminaries: MELD
3. Merging of Decision Diagrams
4. Reasoning about Decision Diagrams
5. Application: DNA Classification
6. Conclusion
Decision Diagrams

Definition (Decision Diagram)

A decision diagram over $D$ and $C$ is a labelled rooted directed acyclic graph

$$D = \langle V, E, \ell_C, \ell_E \rangle$$

- $V$ … nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ … set of directed edges
- $\ell_C : V \rightarrow C$ … partial function assigning a class to all leafs
- $\ell_E : E \rightarrow Q$ … assign queries $Q(z) : D \rightarrow \{true, false\}$ to edges

Query language: $O_1 \circ O_2$ with operands $O_1, O_2$ and $\circ \in \{<, \leq, =, \neq, \geq, >\}$ or “else”

Example

$D = \{1, 2, 3, 4, 5\}$
$C = \{c_1, c_2\}$

[Diagram showing a decision diagram with nodes $v_1, v_2, v_3, v_4$ and nodes $r_D, v_1, v_2, v_3, v_4$, with edges labeled by comparisons and classes $c_1, c_2$.]
Decision Diagrams

Definition (Decision Diagram)

A decision diagram over $D$ and $C$ is a labelled rooted directed acyclic graph

$$D = \langle V, E, \ell_C, \ell_E \rangle$$

- $V$ ... nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ ... set of directed edges
- $\ell_C : V \rightarrow C$ ... partial function assigning a class to all leafs
- $\ell_E : E \rightarrow Q$ ... assign queries $Q(z) : D \rightarrow \{\text{true, false}\}$ to edges

Query language: $O_1 \circ O_2$ with operands $O_1, O_2$ and $\circ \in \{<, \leq, =, \neq, \geq, >\}$ or “else”

Example

$D = \{1, 2, 3, 4, 5\}$
$C = \{c_1, c_2\}$
Classify: 4
Decision Diagrams

Definition (Decision Diagram)

A **decision diagram** over \( D \) and \( C \) is a labelled rooted directed acyclic graph
\[
D = \langle V, E, \ell_C, \ell_E \rangle
\]

- \( V \) … nonempty set of nodes with unique root node \( r_D \in V \)
- \( E \subseteq V \times V \) … set of directed edges
- \( \ell_C : V \rightarrow C \) … partial function assigning a class to all leafs
- \( \ell_E : E \rightarrow Q \) … assign queries \( Q(z) : D \rightarrow \{true, false\} \) to edges

Query language: \( O_1 \circ O_2 \) with operands \( O_1, O_2 \) and \( \circ \in \{<, \leq, =, \neq, \geq, >\} \) or “else”

**Example**

\( D = \{1, 2, 3, 4, 5\} \)
\( C = \{c_1, c_2\} \)
Classify: 4
Decision Diagrams

**Definition (Decision Diagram)**

A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph $D = \langle V, E, \ell_C, \ell_E \rangle$

- $V$ ... nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ ... set of directed edges
- $\ell_C : V \rightarrow \mathcal{C}$ ... partial function assigning a class to all leafs
- $\ell_E : E \rightarrow \mathcal{Q}$ ... assign queries $Q(z) : \mathcal{D} \rightarrow \{true, false\}$ to edges

Query language: $O_1 \circ O_2$ with operands $O_1, O_2$ and $\circ \in \{<, \leq, =, \neq, \geq, >\}$ or “else”

**Example**

$\mathcal{D} = \{1, 2, 3, 4, 5\}$
$\mathcal{C} = \{c_1, c_2\}$
Classify: $4 \Rightarrow c_2$

[Diagram showing a decision tree with labels and queries]
Decision Diagrams

Definition (Decision Diagram)

A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

$$D = \langle V, E, \ell_C, \ell_E \rangle$$

- $V$ ... nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ ... set of directed edges
- $\ell_C : V \rightarrow \mathcal{C}$ ... partial function assigning a class to all leaves
- $\ell_E : E \rightarrow \mathcal{Q}$ ... assign queries $Q(z) : \mathcal{D} \rightarrow \{true, false\}$ to edges

Query language: $O_1 \circ O_2$ with operands $O_1, O_2$ and $\circ \in \{<, \leq, =, \neq, \geq, >\}$ or “else”

Example

$\mathcal{D} = \{1, 2, 3, 4, 5\}$
$\mathcal{C} = \{c_1, c_2\}$
Classify: $4 \Rightarrow c_2$

Note: $\mathcal{D}$ may consist of composed objects, e.g. $Q(z) = z \cdot TSH > 4.5 mU/l$
Decision Diagram Merging

Instantiation of MELD

- How to use MELD for decision diagram merging?
Decision Diagram Merging

Instantiation of MELD

- How to use MELD for decision diagram merging?
  1. Encode decision diagrams as belief sets
  2. Merging by special operators
Decision Diagram Merging

Instantiation of MELD

- How to use MELD for decision diagram merging?
  1. **Encode** decision diagrams as belief sets
  2. **Merging by** special operators

1. Encoding

- **Define nodes**
  \( \text{root}(n), \text{inner}(n), \text{leaf}(n, l) \)

- **Arcs between nodes, labelled with conditions**
  \( \text{cond}(n_1, n_2, o_1, c, o_2), \text{else}(n_1, n_2) \)
1. Encoding of Decision Diagrams

Example

Decision Diagram $D$:

```
E(D) = \{ root(r_D); inner(r_D); inner(v_1); inner(v_2);
        leaf(v_3, c_1); leaf(v_4, c_2);
        cond(r_D, v_1, z, <, 3); else(r_D, v_2);
        cond(v_1, v_3, z, <, 2); else(v_1, v_4);
        cond(v_2, v_3, z, <, 4); else(v_2, v_4) \}
```
2. Merging of Decision Diagrams

Merging

Belief sets = encoded diagrams

\[ \text{BS}(KB_1) \rightarrow \text{BS}(KB_2) \rightarrow \text{BS}(KB_3) \rightarrow \text{BS}(KB_4) \rightarrow \text{BS}(KB_5) \]
2. Merging of Decision Diagrams

Merging

Belief sets = encoded diagrams

\[
\begin{align*}
& \circ X \\
& \circ Y \\
& \circ W \quad E(D_2) \quad E(D_3) \\
& \quad E(D_1) \\
\end{align*}
\]

\[
\begin{align*}
& \circ Z \\
& \quad E(D_4) \quad E(D_5)
\end{align*}
\]
2. Merging of Decision Diagrams

Belief sets = encoded diagrams

Special merging operators $\circ_W$, $\circ_X$, $\circ_Y$, $\circ_Z$ required!
2. Merging of Decision Diagrams

Some Examples of Predefined Operators

- **User Preferences**
  Give some class label preference over another

\[
\circ_{\text{pref}}(D_1, D_2, c_2 > c_1)
\]
2. Merging of Decision Diagrams

Some Examples of Predefined Operators

- **User Preferences**
  Give some class label preference over another

\[ \circ_{\text{pref}} (D_1, D_2, c_2 > c_1) \]

Decision Diagrams:

- \( D_1 \):
  - \( X > 3 \)
  - \( X \leq 3 \)
  - \( c_1 \)
  - \( c_2 \)

- \( D_2 \):
  - \( Y > 2 \)
  - \( Y \leq 2 \)
  - \( c_1 \)
  - \( c_2 \)
2. Merging of Decision Diagrams

Some Examples of Predefined Operators

- **User Preferences**
  Give some class label preference over another

- **Majority Voting**
  Majority of input diagrams decides upon an element’s class

- **Simplification**
  Decrease redundancy

- **MORGAN merging strategy**
  see later

- . . .

**Note:** Operators may produce multiple results!

**Example:** Majority voting for classes with equal number of votes
Outline

1. Motivation
2. Preliminaries: MELD
3. Merging of Decision Diagrams
4. Reasoning about Decision Diagrams
5. Application: DNA Classification
6. Conclusion
Reasoning about Decision Diagrams

Goal

- Compute **diagram properties**
  - e.g. height, variable occurrences, redundancy
- Properties may **control** the **merging process** by **filtering**
Reasoning about Decision Diagrams

Goal

- Compute **diagram properties**
  - e.g. height, variable occurrences, redundancy
- Properties may **control** the merging process by **filtering**

Realization

- Special unary operator
  \[ \circ_{asp}(\Delta, P), \]
  \[ \Delta \ldots \text{set of decision diagrams} \]
  \[ P \ldots \text{ASP program} \]

- \[ P' := P \cup \bigcup_{D \in \Delta} \hat{E}(D) \]
  Extended Encoding \( \hat{E} \):
  *Multiple* diagrams within one set of facts:
  \[ \text{leaf}(L, C) \implies \text{leaf}_{in}(I, L, C) \]

- Evaluate \( P' \) under ASP semantics
Example: Node Count Minimization

\[
P_{\text{min}} = \{ \text{cnt}(I, C) \leftarrow LC = \#\text{count}\{L : \text{leaf}_{\text{in}}(I, L, C)\}, \]
\[
   IC = \#\text{count}\{N : \text{inner}_{\text{in}}(I, N)\}, \]
\[
   \text{root}_{\text{in}}(I, R), C = LC + IC \]
\[
c(I) \leftarrow \text{root}_{\text{in}}(I, R), \neg \text{not } c(I) \]
\[
\neg c(I) \lor \neg c(J) \leftarrow \text{root}_{\text{in}}(I, R), \text{root}_{\text{in}}(J, S), I \neq J \]
\[
\text{leaf}(L, C) \leftarrow c(I), \text{leaf}_{\text{in}}(I, L, C) \]
\[
\ldots \]
\[
\text{else}(N_1, N_2) \leftarrow c(I), \text{else}_{\text{in}}(I, N_1, N_2) \]
\[
\bot \leftarrow M = \#\text{min}\{NC : \text{cnt}(I, NC)\}, \]
\[
c(I), \text{cnt}(I, C), C > M \} \]
Outline

1 Motivation

2 Preliminaries: MELD

3 Merging of Decision Diagrams

4 Reasoning about Decision Diagrams

5 Application: DNA Classification

6 Conclusion
DNA Classification

Motivation

- Given: Sequence over \{A, C, G, T\}
- Question: Is it coding or junk DNA?

Usual Approach

Training

1. Annotated training set
2. Compute statistical features
3. Machine-learning algorithms

Classification

1. Compute the same features
2. Apply decision diagram
DNA Classification

Advanced Approach [Salzberg et al., 1998]

- Train multiple diagrams
  varying training sets, algorithms, features, etc.
- Merge them afterwards

Benefits

- Parallelization
- Increase accuracy (cf. genetic algorithms)
- Smaller training set suffices

Hardcoded implementation: **MORGAN system**
DNA Classification

MORGAN’s strategy in MELD

- MORGAN’s strategy plugged into MELD as merging operator $\circ_M$
- Benefits identified in [5] confirmed

MORGAN vs. MELD-based system

- Not hardcoded but modular
- Clear separation: merging operation / other system components
- reuse / exchange of the merging operator
- Experiment with different merging strategies
- Produce multiple diagrams and reason about them
Outline

1. Motivation
2. Preliminaries: MELD
3. Merging of Decision Diagrams
4. Reasoning about Decision Diagrams
5. Application: DNA Classification
6. Conclusion
Summary

- MELD: Integration of multiple collections of belief sets
- **Instantiation** for decision diagram merging:
  1. Encoding of decision diagrams as belief sets
  2. Special merging operators for decision diagrams
Conclusion

Summary

- MELD: Integration of multiple collections of belief sets
- Instantiation for decision diagram merging:
  1. Encoding of decision diagrams as belief sets
  2. Special merging operators for decision diagrams

Advantages

- Reuse of operators
- Evaluate different operators empirically
- Automatic recomputation of result
- Release user from routine tasks

Download

URL: http://www.kr.tuwien.ac.at/research/dlvhex/ddm.html
References

Dov M. Gabbay, Odinaldo Rodrigues, Gabriella Pigozzi
Connections between Belief Revision, Belief Merging and Social Choice
In: Journal of Logic and Computation 19(3) (2009)

Konieczny, S., Pérez, R.P.:
On the logic of merging.

Redl, C.:
Merging of Biomedical Decision Diagrams
Master’s thesis, Vienna University of Technology (October 2010)

Eiter, T., Ianni, G., Schindlauer, R., Tompits, H.:
dlvhex: A system for integrating multiple semantics in an answer-set programming framework.
In: WLP’06. (2006) 206–210

Salzberg, S., Delcher, A.L., Fasman, K.H., Henderson, J.:
A decision tree system for finding genes in DNA.