Declarative Merging of and Reasoning about Decision Diagrams

Thomas Eiter Thomas Krennwallner Christoph Redl

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

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

kbs
Knowledge-Based Systems Group

September 12, 2011

Outline

Outline

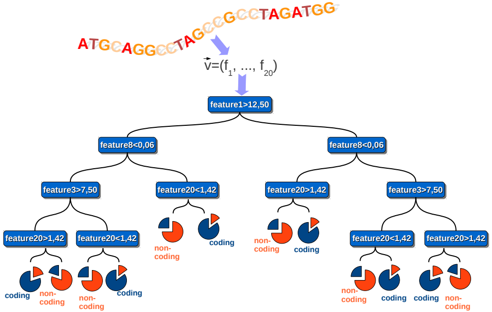
Motivation

Decision Diagrams

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

Examples

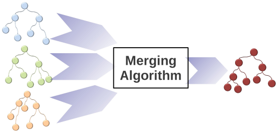
- Severity ratings (e.g. TNM system)
- Diagnosis of personality disorders
- DNA classification



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

MELD

Task

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

Method: Merging Operators

$$\circ^{n,m} : \underbrace{\left(2^{\mathbb{B}_\Sigma}\right)^n}_{collections\ of\ belief\ sets} \times \underbrace{\mathcal{A}_1 \times \ldots \times \mathcal{A}_m}_{operator\ arguments} \rightarrow 2^{\mathbb{B}_\Sigma}$$

Example

Operator definition:

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

Application:

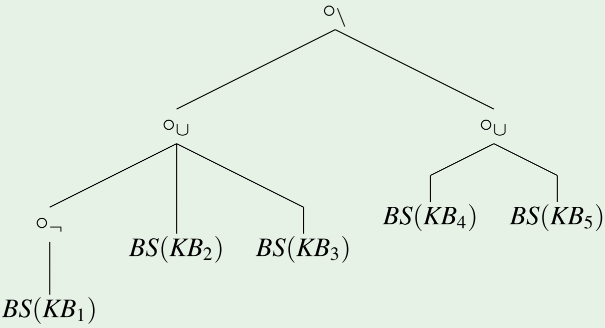
- $\mathcal{B}_1 = \{\{a, b, c\}, \{\neg a, c\}\}, \mathcal{B}_2 = \{\{\neg a, d\}, \{c, d\}\}$
- $\circ_\cup^{2,0}(\mathcal{B}_1, \mathcal{B}_2) = \{\{a, b, c, d\}, \{\neg a, c, d\}\}$

MELD

Merging Plan

- Hierarchical arrangement of merging operators

Example



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

Decision Diagrams

Definition (Decision Diagram)

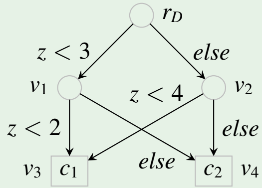
A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

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

- V ... nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ ... set of directed edges
- $\ell_C : V \to \mathcal{C}$ ... partial function assigning a class to all leafs
- $\ell_E : E \to \mathcal{Q}$ ... assign queries $Q(z) : \mathcal{D} \to \{true, false\}$ to edges
  Query language: $O_1 \circ O_2$ with operands O_1, O_2 and $\circ \in \{<, \leq, =, \neq, \geq, >\}$ or "else"

Example

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



Decision Diagrams

Definition (Decision Diagram)

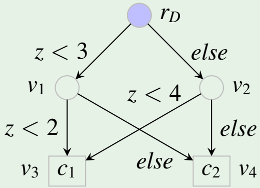
A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

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

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



Decision Diagrams

Definition (Decision Diagram)

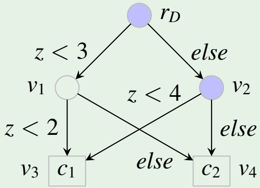
A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

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

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



Decision Diagrams

Definition (Decision Diagram)

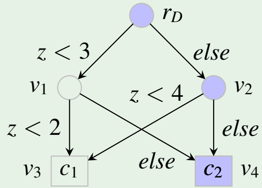
A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

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

- V . . . nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ . . . set of directed edges
- $\ell_C : V \to \mathcal{C}$ . . . partial function assigning a class to all leafs
- $\ell_E : E \to \mathcal{Q}$ . . . assign queries $Q(z) : \mathcal{D} \to \{true, false\}$ 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: $4 \Rightarrow c_2$



Decision Diagrams

Definition (Decision Diagram)

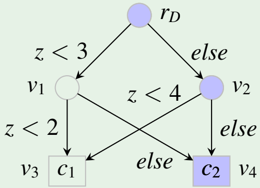
A decision diagram over $\mathcal{D}$ and $\mathcal{C}$ is a labelled rooted directed acyclic graph

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

- V ... nonempty set of nodes with unique root node $r_D \in V$
- $E \subseteq V \times V$ ... set of directed edges
- $\ell_C : V \to \mathcal{C}$ ... partial function assigning a class to all leafs
- $\ell_E : E \to \mathcal{Q}$ ... assign queries $Q(z) : \mathcal{D} \to \{true, false\}$ 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. $Q(z) = z.TSH > 4.5mU/l$

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

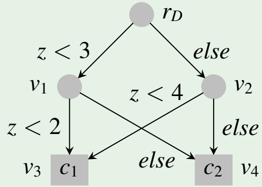
1. Encoding

- Define nodes
  $root(n)$, $inner(n)$, $leaf(n, l)$
- Arcs between nodes, labelled with conditions
  $cond(n_1, n_2, o_1, c, o_2)$, $else(n_1, n_2)$

1. Encoding of Decision Diagrams

Example

Decision Diagram D:

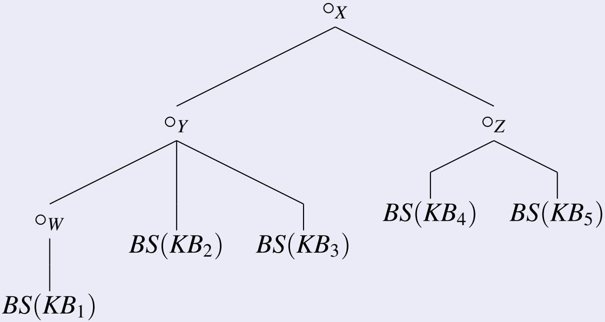


$$\begin{aligned} E(D) = \{ \; & root(r_D); inner(r_D); inner(v_1); inner(v_2); \\ & leaf(v_3, c_1); leaf(v_4, c_2); \\ & cond(r_D, v_1, z, <, 3); else(r_D, v_2); \\ & cond(v_1, v_3, z, <, 2); else(v_1, v_4); \\ & cond(v_2, v_3, z, <, 4); else(v_2, v_4) \} \end{aligned}$$

2. Merging of Decision Diagrams

Merging

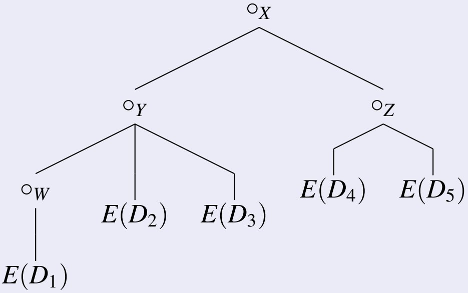
Belief sets = encoded diagrams



2. Merging of Decision Diagrams

Merging

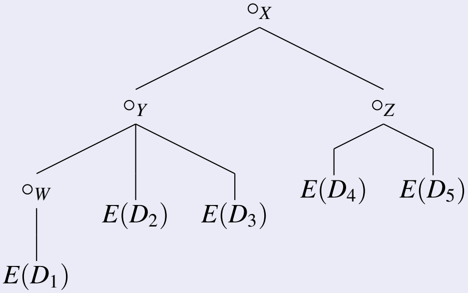
Belief sets = encoded diagrams



2. Merging of Decision Diagrams

Merging

Belief sets = encoded diagrams

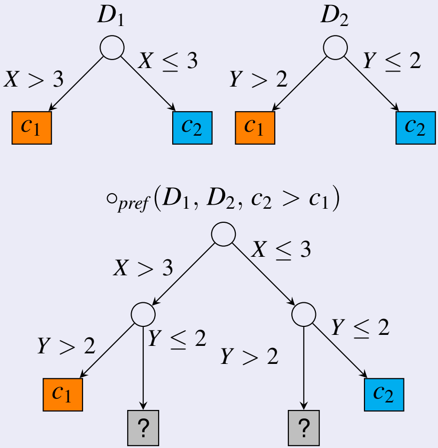


Special merging operators $\circ_W, \circ_X, \circ_Y, \circ_Z$ required!

2. Merging of Decision Diagrams

Some Examples of Predefined Operators

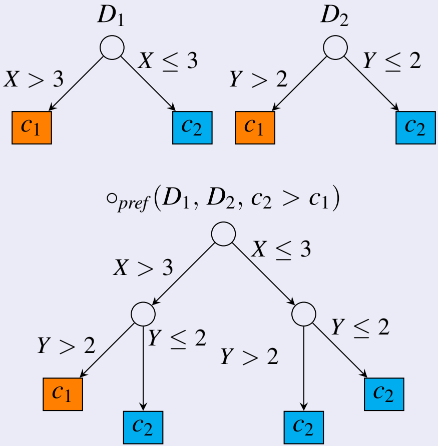
- **User Preferences**
  Give some class label preference over another



2. Merging of Decision Diagrams

Some Examples of Predefined Operators

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



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

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

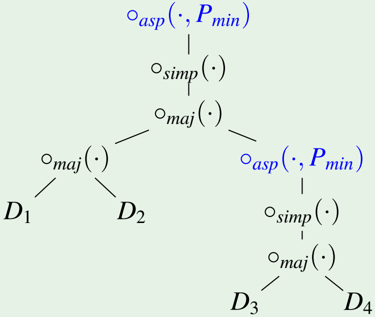
- Special unary operator

$$\circ_{asp}(\Delta, P),$$

  Δ ... set of decision diagrams
  P ... ASP program
- $P' := P \cup \bigcup_{D \in \Delta} \hat{E}(D)$
  Extended Encoding $\hat{E}$:
  Multiple diagrams within one set of facts: $leaf(L, C) \Rightarrow leaf_{in}(I, L, C)$
- Evaluate P' under ASP semantics

Reasoning about Decision Diagrams

Example: Node Count Minimization



$$
\begin{aligned}
P_{min} = \{ cnt(I, C) &\leftarrow LC = \#count\{L : leaf_{in}(I, L, C)\}, \\
&\quad IC = \#count\{N : inner_{in}(I, N)\}, \\
&\quad root_{in}(I, R), C = LC + IC \\
c(I) &\leftarrow root_{in}(I, R), \text{not } \neg c(I) \\
\neg c(I) \vee \neg c(J) &\leftarrow root_{in}(I, R), root_{in}(J, S), I \neq J \\
leaf(L, C) &\leftarrow c(I), leaf_{in}(I, L, C) \\
&\ldots \\
else(N_1, N_2) &\leftarrow c(I), else_{in}(I, N_1, N_2) \\
\bot &\leftarrow M = \#min\{NC : cnt(I, NC)\}, \\
&\quad c(I), cnt(I, C), C > M\}
\end{aligned}
$$

Outline

DNA Classification

Motivation

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

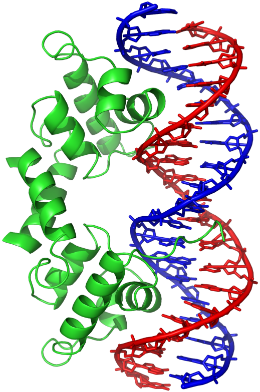
Usual Approach

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

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.
In: KR'98. (1998) 488–498

Redl, C.:
Merging of Biomedical Decision Diagrams
Master's thesis, Vienna University of Technology (October 2010)
`http://www.ub.tuwien.ac.at/dipl/2010/AC07808795.pdf`

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.
Journal of Computational Biology **5**(4) (1998) 667–680