Set Family Decision Diagrams

Dimitri Racordon   Didier Buchs

Centre Universitaire d'Informatique, Université de Genève

December 7, 2017

How to compute efficiently on sets ?

- represent sets in a compact way
- compute on a whole set instead on a single element
  - aka SIMD or *graphic card computing*
- respect union : set homomorphism

various approaches based on decision diagrams.

Set Family Decision Diagrams

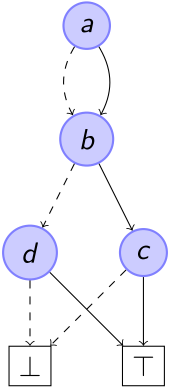
Informal Definition

A SFDD is a directed acyclic graph where

- each node represent a term
- each node has two children, indicating whether or not the term is contained
- each path from the root to an accepting terminal represents a set of terms

Set Family Decision Diagrams

Example



Encodes the sets:

- $\{a, b, c\}$
- $\{a, d\}$
- $\{b, c\}$
- $\{d\}$

Definition (Formal definition)

Let T be a set of terms. The set of SFDDs $\mathbb{S}$ is inductively defined by:

- $\bot \in \mathbb{S}$ is the rejecting terminal
- $\top \in \mathbb{S}$ is the accepting terminal
- $\langle t, \tau, \sigma \rangle \in \mathbb{S}$ if and only if $t \in T \wedge \tau, \sigma \in \mathbb{S}$

Set Family Decision Diagrams

Canonical Form

Let $S \in \mathbb{S}$ be the SFDD $\langle t, \tau, \sigma \rangle$, we call τ its take node and σ its skip node.

S is canonical if for all its nodes, the skip node and take node represent greater terms or terminals, and no take node is the rejecting terminal.

Definition (Canonical form)

Let T be a set of terms, and $< \in T \times T$ a total ordering on T. A SFDD $S \in \mathbb{S}$ is canonical if and only if

- S is the rejecting terminal $\bot$
- S is the accepting terminal $\top$
- $S = \langle t, \tau, \sigma \rangle$ where
  - $\tau = \langle t_\tau, \tau_\tau, \sigma_\tau \rangle \implies t < t_\tau$ and $\tau \neq \bot$
  - $\sigma = \langle t_\sigma, \tau_\sigma, \sigma_\sigma \rangle \implies t < t_\sigma$
  - τ and σ are canonical

Set Family Decision Diagrams
Union

The union of two SFDDs is given by:

$$
\begin{aligned}
A \cup B &= B \cup A \\
A \cup A &= A \\
\bot \cup A &= A \\
\top \cup \langle t, \tau, \sigma \rangle &= \langle t, \top \cup \tau, \top \cup \sigma \rangle \\
\langle t, \tau, \sigma \rangle \cup \left\langle t', \tau', \sigma' \right\rangle &= \begin{cases} \langle t, \tau, \sigma \cup \langle t', \tau', \sigma' \rangle \rangle & \text{if } t < t' \\ \langle t, \tau \cup \tau', \sigma \cup \sigma' \rangle & \text{if } t = t' \\ \langle t', \tau', \sigma' \cup \langle t, \tau, \sigma \rangle \rangle & \text{if } t > t' \end{cases}
\end{aligned}
$$

Set Family Decision Diagrams

Intersection

The intersection of two SFDDs is given by:

$$
\begin{aligned}
A \cap B &= B \cap A \\
A \cap A &= A \\
\bot \cap A &= \bot \\
\top \cap \langle t, \tau, \sigma \rangle &= \top \cap \sigma \\
\langle t, \tau, \sigma \rangle \cap \left\langle t', \tau', \sigma' \right\rangle &= \begin{cases} \sigma \cap \langle t', \tau', \sigma' \rangle & \text{if } t < t' \\ \langle t, \tau \cap \tau', \sigma \cap \sigma' \rangle & \text{if } t = t' \\ \langle t, \tau, \sigma \rangle \cap \sigma' & \text{if } t > t' \end{cases}
\end{aligned}
$$

Set Family Decision Diagrams

Encoding

The encoding of a set into a SFDD is given by:

$$\begin{aligned}
\mathrm{enc}(\varnothing) &= \bot \\
\mathrm{enc}(\{\varnothing\}) &= \top \\
\mathrm{enc}(S \cup \{s\}) &= \mathrm{enc}(S) \cup \mathrm{enc}(\{s\}) \\
t < \min(s) \implies \mathrm{enc}(\{s \cup \{t\}\}) &= \langle t, \mathrm{enc}(\{s\}), \bot \rangle
\end{aligned}$$

Set Family Decision Diagrams
Decoding

The decoding of one SFDD is given by:

$$\begin{aligned}\mathrm{dec}(\bot) &= \varnothing \\ \mathrm{dec}(\top) &= \{\varnothing\} \\ \mathrm{dec}(\langle t, \tau, \sigma \rangle) &= (\mathrm{dec}(\tau) \oplus t) \cup \mathrm{dec}(\sigma)\end{aligned}$$

Where $\oplus$ is defined as follows:

$$\bigcup_{s \in S} \{s\} \oplus t = \bigcup_{s \in S} \{s \cup \{t\}\}$$

The decoding/encoding of one set is the identity (and the reverse):

$$\forall S \subseteq \mathcal{P}(T), \mathrm{dec}(\mathrm{enc}(S)) = S$$
$$\forall S \in \mathbb{S}, \mathrm{enc}(\mathrm{dec}(S)) = S$$

Homomorphisms are operations that preserve union:

$$\phi(S \cup S') = \phi(S) \cup \phi(S')$$

They also support operations that are themselves homomorphisms:

$$\begin{aligned}
\forall S, (\phi_1 + \phi_2)(S) &= \phi_1(S) \cup \phi_2(S) \\
\forall S, (\phi_1 \times \phi_2)(S) &= \phi_1(S) \cap \phi_2(S) \\
\forall S, (\phi_1 \circ \phi_2)(S) &= \phi_1(\phi_2(S))
\end{aligned}$$

Set Family Decision Diagrams

Insertion

$\oplus : \mathbb{S}, T \to \mathbb{S}$ inserts a term $t \in T$ into all sets of a SFDD:

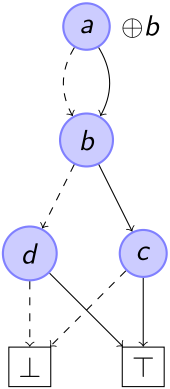
$$
\begin{aligned}
\bot \oplus a &= \bot \\
\top \oplus a &= \langle a, \top, \bot \rangle \\
\langle t, \tau, \sigma \rangle \oplus a &= \begin{cases} \langle t, \tau \oplus a, \sigma \oplus a \rangle & \text{if } t < a \\ \langle t, \tau \cup \sigma, \bot \rangle & \text{if } t = a \\ \langle a, \langle t, \tau, \sigma \rangle, \bot \rangle & \text{if } t > a \end{cases}
\end{aligned}
$$

NB: $\oplus$ is an homomorphism.

Set Family Decision Diagrams

Insertion

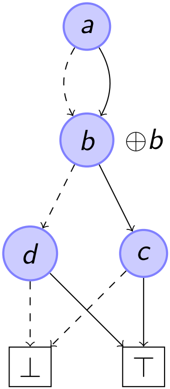
Example: $\mathrm{enc}(\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}) \oplus b$



Set Family Decision Diagrams

Insertion

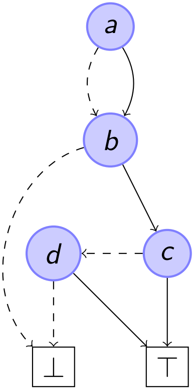
Example: $\mathrm{enc}(\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}) \oplus b$



Set Family Decision Diagrams

Insertion

Example: $\mathrm{enc}(\{\{a,b,c\},\{a,d\},\{b,c\},\{d\}\}) \oplus b$



Encodes the sets:

- $\{a,b,c\}$
- $\{a,b,d\}$
- $\{b,c\}$
- $\{b,d\}$

Set Family Decision Diagrams

Removal

$\ominus : \mathbb{S}, T \to \mathbb{S}$ removes a term $t \in T$ from all sets that contain it:

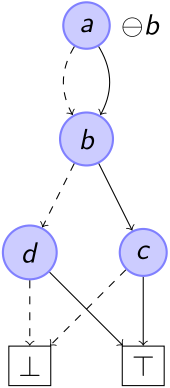
$$\begin{aligned}
\bot \ominus a &= \bot \\
\top \ominus a &= \top \\
\langle t, \tau, \sigma \rangle \ominus a &= \begin{cases} \langle t, \tau \ominus a, \sigma \ominus a \rangle & \text{if } t < a \\ \sigma \cup \tau & \text{if } t = a \\ \langle t, \tau, \sigma \rangle & \text{if } t > a \end{cases}
\end{aligned}$$

NB: $\ominus$ is an homomorphism.

Set Family Decision Diagrams

Removal

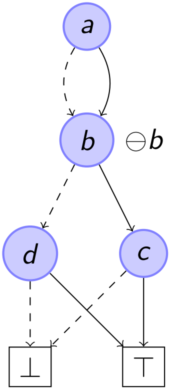
Example: $\mathrm{enc}(\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}) \ominus b$



Set Family Decision Diagrams

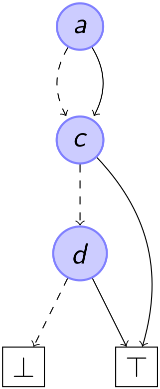
Removal

Example: $\mathrm{enc}(\{\{a,b,c\},\{a,d\},\{b,c\},\{d\}\}) \ominus b$



Set Family Decision Diagrams
Removal

Example: $\mathrm{enc}(\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}) \ominus b$



Encodes the sets:

- $\{a, c\}$
- $\{a, d\}$
- $\{c\}$
- $\{d\}$

Set Family Decision Diagrams

Filtering

filter : $\mathbb{S}, T \to \mathbb{S}$ filters out the sets that don't contain a term $t \in T$:

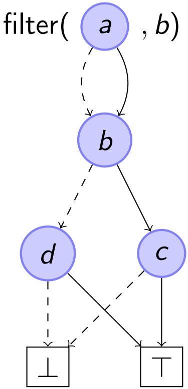
$$
\begin{aligned}
\mathrm{filter}(\bot, a) &= \bot \\
\mathrm{filter}(\top, a) &= \bot \\
\mathrm{filter}(\langle t, \tau, \sigma \rangle, a) &= \begin{cases} \langle t, \mathrm{filter}(\tau, a), \mathrm{filter}(\sigma, a) \rangle & \text{if } t < a \\ \langle t, \tau, \bot \rangle & \text{if } t = a \\ \bot & \text{if } t > a \end{cases}
\end{aligned}
$$

NB1: filter is an homomorphism.

Set Family Decision Diagrams

Filtering

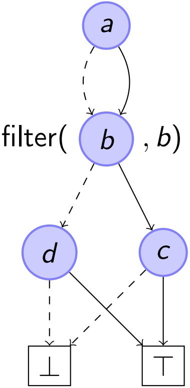
Example: $\text{filter}(\text{enc}(\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}), b)$



Set Family Decision Diagrams

Filtering

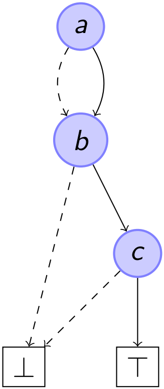
Example: filter(enc($\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}$), b)



Set Family Decision Diagrams

Filtering

Example: filter(enc($\{\{a, b, c\}, \{a, d\}, \{b, c\}, \{d\}\}$), b)



Encodes the sets:

- $\{a, b, c\}$
- $\{b, c\}$

An inductive homomorphism is a tuple $\phi = \langle S, i \rangle$ where:

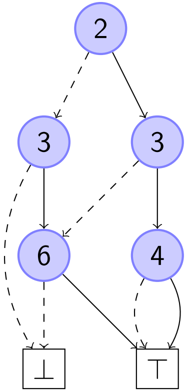
- $S \in \mathbb{S}$
- $i(A) = \langle \phi_\tau, \phi_\sigma \rangle$ where ϕ_τ, ϕ_σ are homomorphisms and $A \in \mathbb{S} \setminus \{\bot, \top\}$

Let $\phi = \langle S, i \rangle$, its application on $A \in \mathbb{S}$ is given by:

$$\phi(A) = \begin{cases} \bot & \text{if } A = \bot \\ S & \text{if } A = \top \\ \langle t, \phi_\tau(\tau), \phi_\sigma(\sigma) \rangle & \text{if } A = \langle t, \tau, \sigma \rangle, i(A) = \langle \phi_\tau, \phi_\sigma \rangle \end{cases}$$

Set Family Decision Diagrams

Inductive Homomorphisms

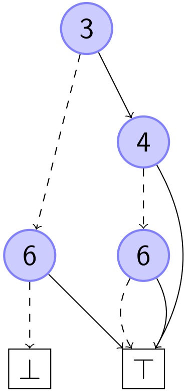


Example: removing values smaller than of 4

$$\phi = \langle \top, i \rangle$$

$$i(\langle t, \tau, \sigma \rangle) = \begin{cases} \langle h[\bot], \phi \circ (h[\tau] + \mathrm{id}) \rangle & \text{if } t < 4 \\ \langle \mathrm{id}, \mathrm{id} \rangle & \text{otherwise} \end{cases}$$

where $\forall S, \mathrm{id}(S) = S$ and $\forall S, h[K](S) = K$.

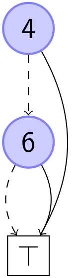


Example: removing values smaller than of 4

$$\phi = \langle \top, i \rangle$$

$$i(\langle t, \tau, \sigma \rangle) = \begin{cases} \langle h[\bot], \phi \circ (h[\tau] + \mathrm{id}) \rangle & \text{if } t < 4 \\ \langle \mathrm{id}, \mathrm{id} \rangle & \text{otherwise} \end{cases}$$

where $\forall S, \mathrm{id}(S) = S$ and $\forall S, h[K](S) = K$.



Example: removing values smaller than of 4

$$\phi = \langle \top, i \rangle$$

$$i(\langle t, \tau, \sigma \rangle) = \begin{cases} \langle h[\bot], \phi \circ (h[\tau] + \mathrm{id}) \rangle & \text{if } t < 4 \\ \langle \mathrm{id}, \mathrm{id} \rangle & \text{otherwise} \end{cases}$$

where $\forall S, \mathrm{id}(S) = S$ and $\forall S, h[K](S) = K$.

The count of members in a family is the homomorphism size:

$$\mathrm{size}(S) = \begin{cases} 0 & \text{if } S = \bot \\ 1 & \text{if } S = \top \\ \mathrm{size}(\tau) + \mathrm{size}(\sigma) & \text{if } S = \langle t, \tau, \sigma \rangle \end{cases}$$

NB1: size is an homomorphism.

Set Family Decision Diagrams

Algorithm 1: State space computation on individual states

Input: s_0 : initial state.
Input: T : set of transition.
Result: set of reachable states

```
begin
    s_rem, s : set of states ;
    m, mt : states ;
    s_rem ← {s_0} ; s ← {};
    repeat
        m ← choose(s) ;
        s_rem ← s_rem/{m} ;
        foreach t ∈ T do
            if fireable(t,m) then
                mt ← t(m) ;
                if m ∉ s then s ← s ∪ {mt}; s_rem ← s_rem ∪ {mt};
    until s_rem = ∅;
    return s;
```

Algorithm 2: Global state space computation

Input: s_0 : initial state.
Input: Φ : set of transition homomorphisms.
Result: set of reachable states

```
begin
    s, s_old, temp : set of states ;
    s ← {s_0} ;
    repeat
        s_old ← s ;
        foreach t ∈ Φ do
            temp ← t(s) ;
            s ← s ∪ temp ;
    until s = s_old;
    return s;
```

What about $t(s)$?

$$t(m) = m + post(t) - pre(t)$$

If *pre* and *post* are functions on transition and markings.

$$t(m) = post(t, pre(t, m))$$

Extended to set of states:

$$t(s \cup \{m\}) = t(t(s) \cup \{post(t, pre(t, m))\})$$
$$t(\varnothing) = \varnothing$$

Petri nets are defined as $\langle P, T, Pre, Post\rangle$ where:

- P and T are finite disjoint sets.
- Pre and $Post$ are functions $P \times T \rightarrow \mathbb{N}$

The state of a Petri net is the marking $M : P \rightarrow \mathbb{N}$.

A transition $t \in T$ is fireable if and only if

$$\forall p \in P, Pre(p, t) \leq M(p)$$

The firing of a transition modifies the marking (i.e. state):

$$\forall p \in P, M'(p) = M(p) + Post(p, t) - Pre(p, t)$$

Encoding a safe Petri net marking M with S_M can be done with sets using simply P as terms:

$$S_M = \bigcup_{p \in P, M(p)=1} \{p\}$$

which is encoded directly in SFDD as: $enc(\bigcup_{p \in P, M(p)=1} \{p\})$ with the total order $P = \{p_1, p_2, ..., p_k\}$ and $p_1 < p_2 < \cdots < p_k$

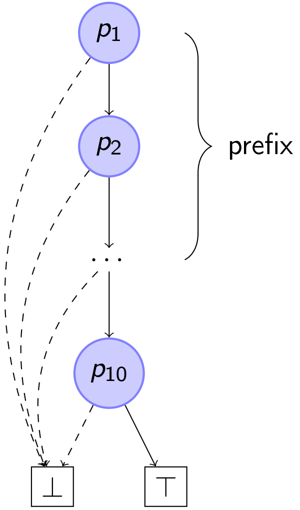
Set Family Decision Diagrams

Encoding safe PN pre and post conditions

$$
\begin{aligned}
t &= post(t) \circ pre(t) \\
pre(t) &= pre(t, p_1) \circ pre(t, p_2) \circ \cdots \circ pre(t, p_n) \\
post(t) &= post(t, p_1) \circ post(t, p_2) \circ \cdots \circ post(t, p_1) \\
pre(t, p_i) &= \begin{cases} \ominus(p_i) \circ \mathrm{filter}(p_i) & \text{if } Pre(t, p_i) \neq 0 \\ (id) & \text{otherwise} \end{cases} \\
post(t, p_i) &= \begin{cases} \oplus(p_i) & \text{if } Post(t, p_i) \neq 0 \\ (id) & \text{otherwise} \end{cases}
\end{aligned}
$$

Homomorphisms may involve uncessary operations on large prefixes:

$$\text{filter}(p_{10})(\text{enc}(\bigcup_{1 \leq i \leq 10} p_i))$$



The idea is to dive as deep as possible before applying an homomorphism:

$$\mathrm{dive}(k, \phi)(\bot) = \bot$$

$$\mathrm{dive}(k, \phi)(\top) = \top$$

$$\mathrm{dive}(k, \phi)(\langle t, \tau, \sigma \rangle) = \begin{cases} \langle t, \mathrm{dive}(k, \phi)(\tau), \mathrm{dive}(k, \phi)(\sigma) \rangle & \text{if } t < k \\ \phi(\langle t, \tau, \sigma \rangle) & \text{if } t = k \\ \langle t, \tau, \sigma \rangle & \text{if } t > k \end{cases}$$

Grouping homomorphisms that work on close variables can avoid processing long prefixes multiple times:

$$\mathsf{filter}(p_8) \circ \mathsf{filter}(p_{10}) \equiv \mathrm{dive}(p_8, \mathsf{filter}(p_8) \circ \mathsf{filter}(p_{10}))$$

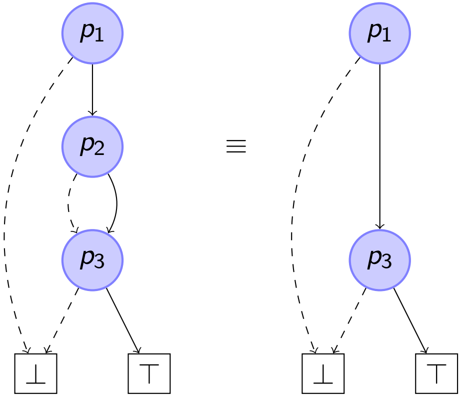
Some homomorphism may be reordered so they can be grouped:

$$\mathsf{filter}(p_i) \circ \mathsf{filter}(p_j) \equiv \mathsf{filter}(p_j) \circ \mathsf{filter}(p_i)$$

Set Family Decision Diagrams

Optimizations

If set of terms is sparse and has a minimum value $t \in T$ such that $\forall u \in T, t \neq u \implies t < u$, some nodes can be omitted:



- SFDD encoding of sets
- SFDD properties such as canonization
- Homomorphic operations on SFDD
- Inductive homomorphisms as pattern of computation
- Encoding of markings and set of markings
- Encoding of fire functions
- Computation of PN state space
- *don't care* can be defined for particular constraint on T