1 GroBdd

1.1 Initial definition of GroBdd

A GroBdd is very similar to an actual ROBDD, the two main differences being that (1) it represents a vector of Boolean functions with a finite number of variables possibly of different arity and (2) on every edge their is a transformation (the "output inverter" is an example of such transformations").

A Reduced Ordered Binary Decision Diagram is a directed acyclic graph $(V \cup T, \Psi \cup E)$ representing a vector of Boolean functions $F = (f_1, ..., f_k)$. Nodes are partitioned into two sets : the set of internal nodes V and the set of terminal nodes T. Every internal node $v \in V$ has two outgoing edges respectively denoted $if0$ and $if1$. Every internal node $v \in V$ has a field *index* which represent a unique identifier associated to each node. Arcs are partitioned into two sets : the set of root arcs Ψ and the set of internal arcs E. There is exactly k root arcs, a root arc is denoted Ψ_i with $0 \leq i < k$, informally, Ψ_i is the root of the GroBdd representing f_i. Every arc has a transformation descriptor field γ and a destination node denoted *node*.

We denote $\rho(\gamma) : \mathbb{F}_n \longrightarrow F_m$ the semantic interpretation of the transformation descriptor γ. We define $\phi(node)$ the semantic of the node *node* and $\psi(arc)$ the semantic of the arc *arc* as follow:

- $\forall i, f_i = \psi(\Psi_i)$
- $\forall arc \in \Psi \cup E, \psi(arc) = \rho(arc.\gamma)(\phi(arc.node))$
- $\forall node \in V, \phi(node) = \psi(node.if0) \star \psi(node.if1)$

We assume the function ϕ defined on all terminals T (we always assume, all terminal to have a different interpretation through ϕ).

1.2 Transformation Descriptor Set (TDS)

We denote $\mathbb{Y}$ the set of all transformation descriptor. We assumme defined ρ the semantical interpretation of transformation descriptors, $\forall \gamma, \exists n, m \in \mathbb{N}, \rho(\gamma) \in \mathbb{F}_n \to \mathbb{F}_m$ (with $n \leq m$). In this section, we make the list of properties that the TDS must ensure to be correct.

1.2.1 Canonical

$$\forall \gamma, \gamma' \in \mathbb{Y}, (\gamma = \gamma') \Leftrightarrow (\rho(\gamma) = \rho(\gamma'))$$

We denote $\mathbb{Y}_{n,m} = \{\gamma \in \mathbb{Y} \mid \rho(\gamma) \in \mathbb{F}_n \longrightarrow \mathbb{F}_m\}$, and $\mathbb{Y}_{n,*} = \bigcup_{m \leq n} \mathbb{Y}_{n,m}$ and $\mathbb{Y}_{*,m} = \bigcup_{n \geq m} \mathbb{Y}_{n,m}$.

1.2.2 Separable

$$\forall \gamma \in \mathbb{Y}_{n,m}, \exists!\ \triangle_\gamma \in \mathbb{B}^m \to \mathbb{B}^n, \triangledown_\gamma \in \mathbb{B}^m \to \mathbb{B} \to \mathbb{B},$$
$$\forall f \in \mathbb{F}_n, \forall x \in \mathbb{B}^m, \rho(\gamma)(f)(x) = \triangledown(x, f(\triangle\ x))$$

This constraint allows any transformation to be represented as circuit which can be wrapped around the function. This constraint enforces transformations to be local. The function $\triangle$ is called the pre-process, and the function $\triangledown$ is called the post-process.

1.2.3 Composable (definition of `C`)

$$\forall \gamma \in \mathbb{Y}_{n,m}, \gamma' \in \mathbb{Y}_{m,l}, \exists \gamma'' \in \mathbb{Y}_{n,l}, \rho(\gamma') \circ \rho(\gamma) = \rho(\gamma'')$$

This constraint enforces $\mathbb{Y}_{n,n}$ to be stable by composition. Furthermore, it exists an algorithm $\mathtt{C} : \mathbb{Y}_{n,m} \to \mathbb{Y}_{m,l} \to \mathbb{Y}_{n,l}$, such that:

$$\forall \gamma \in \mathbb{Y}_{n,m}, \gamma' \in \mathbb{Y}_{m,l}, \rho(\gamma') \circ \rho(\gamma) = \rho(\mathtt{C}(\gamma, \gamma'))$$

For convenience, we denote $\gamma' \circ \gamma = \mathtt{C}(\gamma, \gamma')$.

1.2.4 Decomposable (definition of `A` and `S`)

For all $n \in \mathbb{N}$, we define $A_n = \mathbb{Y}_{n,n}$ the set of asymmetric transformations. For all $n, m \in \mathbb{N}$, it exists $S_{n,m} \subset \mathbb{Y}_{n,m}$ a set of transformations such that $\forall \gamma \in \mathbb{Y}_{n,m}, \exists a \in A_m, \exists! s \in S_{n,m}, \gamma = a \circ s$. The set $S_{n,m}$ is called the set of symmetric transformation.

definition : S-free
A Boolean function $f \in \mathbb{F}_m$ is said S-free, iff

$$\forall s \in S_{n,m}, \forall g \in \mathbb{F}_n, f = \rho(s)(g) \Rightarrow \rho(s) = Id$$

constraint : S-uniqueness

$$\forall f \in \mathbb{F}_n, f' \in \mathbb{F}_{n'}, s \in S_{n,m}, s' \in S_{n',m}, \rho(s)(f) = \rho(s')(f') \Rightarrow (s = s') \wedge (f = f')$$

definition : A-equivalent
Two Boolean functions $f, g \in \mathbb{F}_n$ are said A-equivalent, iff $\exists a \in A_n, f = \rho(a)(g)$. This relation is an equivalence relation (i.e. reflexive, symmetric, transitive) denote $\sim_A$.

definition : A-invariant free
A Boolean function $f \in \mathbb{F}_n$ is said A-invariant free, iff $\forall a, a' \in A_n, \rho(a)(f) \neq \rho(a')(f)$.

constraint : S-free implies A-invariant free
For all function f, if f is S-free, then f is A-invariant free.

definition : A-reduced set of function
Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of Boolean functions. The set X is said A-reduced iff $\forall x_i, x_j \in X, (x_i \sim_A x_j) \Rightarrow (x_i = x_j)$.

definition : $\mathbb{I}_n$
For all $n \in \mathbb{N}$, we denote $\mathbb{I}_n$ the set of identifiers corresponding to node representing functions of arity n.

definition : $\mathbb{Y}-$node

A $\mathbb{Y}-\texttt{node}_{l,m,n}$ is a quadruple $(\gamma_0, I_0, \gamma_1, I_1) \in \mathbb{Y}_{l,n} \times \mathbb{I}_l \times \mathbb{Y}_{m,n} \times \mathbb{I}_m$. Let v be a $\mathbb{Y}-\texttt{node}$, we denote $v.\gamma_0$ (respectively $v.I_0$, $v.\gamma_1$ and $v.I_1$) the first (respectively second, third and fourth) component of v.

- For all $\mathbb{Y}-\texttt{node}$ v, we always assume that functions $\phi(v.I_0)$ and $\phi(v.I_1)$ are S-free.
- We always assume the set $X = \{\phi(v.I_0) \mid v \in \mathbb{Y}-\texttt{node}\} \cup \{\phi(v.I_1) \mid v \in \mathbb{Y}-\texttt{node}\}$ to be A-reduced

We extend the definition of ϕ, with : for all $\mathbb{Y}-\texttt{node}$ v, $\phi(v) = \rho(v.\gamma_0)(\phi(v.I_0)) \star \rho(v.\gamma_1)(\phi(v.I_1))$.

constraint : terminal nodes are S-free and A-reduced (definition of $\texttt{E}_0$) For all terminal node $t \in T$, t is S-free. Furthermore, the set $\{\phi(t) \mid t \in T\}$ is A-reduced. Moreover, we define the function $\texttt{E}_0$ such $\forall b \in \mathbb{F}_0, \exists \gamma \in \mathbb{Y}_{0,0}, \exists t \in T, \texttt{E}_0(b) = \{\gamma = \gamma, node = I_t\}$ and $\psi(\texttt{E}_0(b)) = b$.

1.2.5 Buildable (definition of B)

Let X be a function, we denote I_X the identifier of an hypothetical node whose semantic interpretation is X.

We define B an algorithm over $\mathbb{Y}$ which respect the signature:

$$\begin{aligned}
&\texttt{B} : \mathbb{Y}_{n_0,m} \times \mathbb{I}_{n_0} \longrightarrow \mathbb{Y}_{n_1,m} \times \mathbb{I}_{n_1} \longrightarrow \\
&\quad \mid \ \text{ConsNode} \ \ \mathbb{Y}_{n',m} \times (\mathbb{Y}_{n_x,n'} \times \mathbb{I}_{n_x}) \times (\mathbb{Y}_{n_y,n'} \times \mathbb{I}_{n_y}) \\
&\quad \mid \ \text{Merge} \ \ \mathbb{Y}_{n_z,m} \times \mathbb{I}_{n_z}
\end{aligned}$$

(with $x, y, z \in \{0, 1\}$)
Furthermore, for all $(\gamma_g, I_g, \gamma_h, I_h) \in \mathbb{Y}_{n_0,m} \times \mathbb{I}_{n_0} \times \mathbb{Y}_{n_1,m} \times \mathbb{I}_{n_1}$,

$$\texttt{B}(\gamma_g, I_g, \gamma_h, I_h) = \texttt{ConsNode}(\gamma, (\gamma', I_X), (\gamma'', I_Y)) \Rightarrow f = \rho(\gamma)\left(\rho(\gamma')(X) \star \rho(\gamma'')(Y)\right)$$

$$\texttt{B}(\gamma_g, I_g, \gamma_h, I_h) = \texttt{Merge}(\gamma''', I_Z) \Rightarrow f = \rho(\gamma''')(Z)$$

(with $X, Y, Z \in \{g, h\}$ and $f = \rho(\gamma_g)(g) \star \rho(\gamma_h)(y)$)

definition : a node is B-stable

Let G be a GroBdd, we denote v an internal node of G. We denote $\gamma_0 = v.if0.\gamma, I_0 = v.if0.node, \gamma_1 = v.if1.\gamma$ and $I_1 = v.if1.node$. The node v is said B-stable iff $\texttt{B}(\gamma_0, I_0, \gamma_1, I_1) = \texttt{ConsNode}(Id, (\gamma_0, I_0), (\gamma_1, I_1))$.

constraint : B is B-stable

$$\begin{gathered}
\forall \gamma_f, I_f, \gamma_g, I_g, \texttt{B}(\gamma_f, I_f, \gamma_g, I_g) = \texttt{ConsNode}(\gamma, (\gamma_0, I_0), (\gamma_1, I_1)) \\
\Rightarrow \texttt{B}(\gamma_0, I_0, \gamma_1, I_1) = \texttt{ConsNode}(Id, (\gamma_0, I_0), (\gamma_1, I_1))
\end{gathered}$$

Informally, when B returns a node, this node is B-stable.

constraint : $\mathbb{Y}-$node are B-stable

1. We assume all $\mathbb{Y}-\texttt{node}$ v to be B-stable

$$\texttt{B}(v.\gamma_0, v.I_0, v.\gamma_1, v.I_1) = \texttt{ConsNode}(Id, (v.\gamma_0, v.I_0), (v.\gamma_1, v.I_1))$$

constraint : B is S-free preserving
The algorithm `B` is said S-free preserving iff for all $\mathbb{Y}-\texttt{node}$ v, $\phi(v)$ is S-free.

constraint : B is A-reduction preserving
The algorithm `B` is said A-reduction preserving iff

$$\forall v, w \in \mathbb{Y}-\texttt{node}, \phi(v) \sim_A \phi(w) \Rightarrow v = w$$

1.3 Reduction Rules

We define a GroBdd model as the triple $(\mathbb{Y}, \texttt{C}, \texttt{B})$. A valid model must satisfy all the previously mentioned constraints.

In addition to the previous constraints, we define two reduction rules:

1. The syntactical reduction : all sub-graphs are different up to graph-isomorphism (i.e. all identical sub-graphs are merged)
2. The local semantic reduction : all internal node $v \in V$ is `B`-stable.
3. All node has at least one incoming arc. A GroBdd is said reduced if it satisfies the reduction rules.

In this section we prove:

1. For all vector of Boolean function F it exists a reduced GroBdd G representing it.
2. A reduced GroBdd G is semi-canonical, defined as :
   (a) For all node $v \in V$, $\phi(v)$ is S-free.
   (b) The set $X = \{\phi(v_1), \ldots, \phi(v_N)\}$ representing the set of the semantic interpretation of the set of internal nodes V is A-reduced.
   (c) $\forall(\gamma, I, \gamma', I') \in (\mathbb{Y}_{n,m} \times \mathbb{I}_n)^2, \psi(\{\gamma = \gamma, node = I\}) = \psi(\{\gamma = \gamma', node = I'\}) \Rightarrow (\gamma, I) = (\gamma', I')$ (with I and I' being indexes of nodes in G).
3. Between two reduced GroBdd G and G' representing the same vector of Boolean functions F, it exists a one-to-one mapping $\sigma : V \longrightarrow V'$ such that $\forall v, v' \in V \times V', \sigma(v) = v' \Rightarrow (\exists a \in A_*, \phi(v) = \rho(a)(\phi(v')))$.

1.3.1 Additional procedures

The `Cons` procedure
We define the `Cons` procedure as follow. Let G be a GroBdd and F be its vector of root arcs of size k.
Let i and j be indexes of this vector. We denote $\gamma_i = \Psi_i.\gamma$, $\gamma_j = \Psi_j.\gamma$ and $I_i = \Psi_i.node$, $I_j = \Psi_j.node$.

- If $\texttt{B}(\gamma_g,\ I_g,\ \gamma_h,\ I_h) = \texttt{ConsNode}\ (\gamma,\ (\gamma',\ I_X),\ (\gamma'',\ I_Y))$. We define the node $N = \{if0 = \{\gamma = \gamma', node = I_X\}, if1 = \{\gamma = \gamma'', nodes = I_Y\}\}$.
  - If the node N already exists in G, we retrieve its identifier I, we define G' as a copy of G with a new root arc $\Psi_k = \{\gamma = \gamma, node = I\}$

- Otherwise, we define G' as a copy of G with add N to the list of nodes of G' and create it a new identifier I, we add a new root arc $\Psi_k = \{\gamma = \gamma, node = I\}$.

- Otherwise, $\mathtt{B}(\gamma_g,\ I_g,\ \gamma_h,\ I_h) = \mathtt{Merge}\ (\gamma,\ I_Z)$. We define G' as a copy of G with a new root arc $\Psi_k = \{\gamma = \gamma''', node = I_Z\}$.

The `Remove` procedure

We define the `Remove` procedure as follow. Let G be a GroBdd and F be its vector of root arc of size k.
Let i be an index of this vector We define G' as a copy of G with its vector of root arc $F' = (\Psi_1, \ldots, \Psi_{i-1}, \Psi_{i+1}, \ldots, \Psi_n)$. In an iterative process, we remove nodes which have no incoming arc, until all nodes have at least one incoming arc (in order to satisfy the third reduction rule). As G' is a subset of G, thus, G' satisfies the first and second reduction rule.

The `Reduction` procedure

Let G be a GroBdd and F be its vector of root arcs F. Using an iterative process, we remove nodes which have no incoming arc, until all nodes have at least one incoming arc (satisfying the third reduction rules). The `Reduction` procedure consist in going through the GroBdd G (starting with nodes with the smallest depth), applying the `Cons` procedure on each node creating a new GroBdd G'. The GroBdd G' is by construction equivalent to G, however, the GroBdd G' satisfies all three reduction rules.

The `Merge` procedure

We define the `Merge` procedure as follow. Let G and G' be two GroBdds (based on the same GroBdd model). We define G'' a GroBdd which is the union of both GroBdd. We define $F'' = (\Psi_1, \ldots, \Psi_n, \Psi'_1, \ldots, \Psi'_{n'})$. Due to possible conflicts on identifiers, we re-generate identifiers of the nodes in G''. Finally, we apply the `Reduction` procedure on G''.

1.3.2 Existence

We inductively define the procedure `E` with:

- $\forall b \in \mathbb{F}_0, \mathtt{E}(b) = \mathtt{E}_0(b)$

- Let f be a Boolean function of arity n (with $n \geq 1$). Let G be a GroBdd representing functions f_0 (the negative restriction of f according to its first variable) and f_1 (the positive restriction of f according to its first variable.) by using the procedure `E` on f_0 and f_1. Let G' be the output of the `Cons` procedure on G, in order to create $f = f_0 \star f_1$ (expansion theorem).
  $E(f) = G'$ The GroBdd G' satisfies the reduction rules:

  - If no node is created, the proof is straightforward.
  - If a node is created, this node is syntactically unique by definition of `Cons` and is `B`-stable (as `B` is `B`-stable)

By construction, the procedure `E` (generalized to accept a vector of function as input) returns a GroBdd satisfying the reduction rules.

1.3.3 Semi-Canonical

Let G be a reduced GroBdd.

S-free and A-reduced For all node $v \in V$, we denote $h(v) = max(h(v.if0.node), h(v.if1.node))$ with $\forall t \in T, h(t) = 0$. For all $n \in \mathbb{N}$, we define $V_n = \{v \in V \mid h(v) \leq n\}$ For all $n \in \mathbb{N}$, we define the recurrence hypothesis $H(n)$:

- For all $v \in V_n$, $\phi(v)$ is S-free.
- The set of Boolean function $X = \{\phi(v) \mid v \in V_n\}$ is A-reduced.

Initialization We prove $H(0)$ using the constraints that (1) terminals are S-free and (2) the set of terminal nodes is A-reduced.

Induction Let $n \in \mathbb{N}$, we assume $\forall k \leq nH(k)$. Let v be a node of depth $n+1$, thus the depth of $v.if0.node$ and $v.if1.node$ is lower than n (we can apply the recurrence hypothesis). Therefore, the quadruple $\bar{v} = (v.if0.\gamma, v.if0.node, v.if1.\gamma, v.if1.node)$ is a $\mathbb{Y}-$`node`. Thus, using the constraints that `B` is S-free preserving, we prove that $\phi(v) = \phi(\bar{v})$ is S-free. Let v' be a node of depth $k \leq n+1$, we can prove that the quadruple $\bar{v'} = (v'.if0.\gamma, v'.if0.node, v'.if1.\gamma, v'.if1.node)$ is a $\mathbb{Y}-$`node`. Therefore, we can use the constraint that `B` is A-reduction preserving to prove that $\phi(\bar{v}) \sim_A \phi(\bar{v'}) \Rightarrow \phi(\bar{v}) = \phi(\bar{v'})$ However, $\phi(v) = \phi(\bar{v})$ and $\phi(v') = \phi(\bar{v'})$ Thus, $\forall v, w \in V_{n+1}, \phi(v) \sim_A \phi(v') \Rightarrow \phi(v) = \phi(v')$. Thus, the set of Boolean function $X = \{\phi(v) \mid v \in V_{n+1}\}$ is A-reduced. Therefore $(land_{k \leq n} H(k) \Rightarrow H(n+1)$.

Using the strong recurrence theorem, we prove that $\forall n \in \mathbb{N}, H(n)$. Therefore proving properties (2.a) "all nodes are S-free" and (2.b) "the set $X = \{\phi(v_1), \ldots, \phi(v_N)\}$ is A-reduced".

Semantic Reduction We prove the property "$\forall (\gamma, I, \gamma', I') \in (\mathbb{Y}_{n,m} \times \mathbb{I}_n)^2, \psi(\{\gamma = \gamma, node = I\}) = \psi(\{\gamma = \gamma', node = I'\}) \Rightarrow (\gamma, I) = (\gamma', I')$ (with I and I' being indexes of nodes in G)" by induction on $n \in \mathbb{N}$ the arity of $f = \psi(\{\gamma = \gamma, node = I\})$.

Initialization The induction property holds for $n = 0$:
Let $f \in \mathbb{F}_0$, we assume it exists a quadruple $(\gamma, I, \gamma', I') \in (\mathbb{Y}_{n,m} \times \mathbb{I}_n)^2$ such that $f = \psi(\{\gamma = \gamma, node = I\}) = \psi(\{\gamma = \gamma', node = I'\})$. We decompose γ and γ' to their symmetric and asymmetric components: $\gamma = s \circ a$ and $\gamma' = s' \circ a'$. Using the S-uniqueness constraint, on $f = \rho(s)(\rho(a)(\phi(I))) = \rho(s')(\rho(a')(\phi(I')))$, we have $s = s'$ and $\rho(a)(\phi(I)) = \rho(a')(\phi(I'))$. Therefore, $\phi(I) \sim_A \phi(I')$, however, we proved that the set of the semantic interpretations of the nodes is A-reduced, thus $\phi(I) = \phi(I')$ However, the only node representing function of arity 0 are terminal nodes, therefore $I = I'$.

Induction Let $k \in \mathbb{N}$, we assume the induction property holds for all $n \leq k$, let prove it holds for $n = k+1$. Let f be a Boolean function, we assume it exists a quadruple $(\gamma, I, \gamma', I') \in (\mathbb{Y}_{n,m} \times \mathbb{I}_n)^2$ such that $f = \psi(\{\gamma = \gamma, node = I\}) = \psi(\{\gamma = \gamma', node = I'\})$. We decompose γ and γ' to their symmetric and asymmetric components: $\gamma = s \circ a$ and $\gamma' = s' \circ a'$. Using the

S-uniqueness constraint, on $f = \rho(s)(\rho(a)(\phi(I))) = \rho(s')(\rho(a')(\phi(I')))$, we have $s = s'$ and $\rho(a)(\phi(I)) = \rho(a')(\phi(I'))$. Therefore, $\phi(I) \sim_A \phi(I')$, however, we proved that the set of the semantic interpretations of the nodes is A-reduced, thus $\phi(I) = \phi(I')$ Using the induction hypothesis (as $\phi(I)$ as an arity strictly smaller than $k+1$, thus smaller than k), we deduce that $I = I'$.

Therefore, applying the strong induction theorem, we prove the property (2.c).

1.3.4 Canonical modulo graph-isomorphism and A-equivalence

In order to prove that "Between two reduced GroBdd G and G' representing the same vector of Boolean functions F, it exists a one-to-one mapping $\sigma : V \longrightarrow V'$ such that $\forall v, v' \in V \times V', \sigma(v) = v' \Rightarrow (\exists a \in A_*, \phi(v) = \rho(a)(\phi(v'))$", we start by (1) proving that within the `Merge` procedure, each node is replace by an A-equivalent one then (2) using the `Merge` procedure without re-generating the identifiers of the nodes of G and using the property (2.c), we prove that the nodes of G' collapse on the nodes of G during the `Reduction` procedure, providing a one-to-one mapping. Details of this proof are cumbersome and not of great interest