# Minimization of P-Circuits using Boolean Relations

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
annab@di.unipi.it

Valentina Ciriani, Gabriella Trucco
Departimento di Informatica
Università degli Studi di Milano, Italy
{valentina.ciriani, gabriella.trucco}@unimi.it

Tiziano Villa
Departimento di Informatica
Università degli Studi di Verona, Italy
tiziano.villa@univr.it

***Abstract*—In this paper, we investigate how to use the complete flexibility of P-circuits, which realize a Boolean function by projecting it onto overlapping subsets given by a generalized Shannon decomposition. It is known how to compute the complete flexibility of P-circuits, but the algorithms proposed so far for its exploitation do not guarantee to find the best implementation, because they cast the problem as the *minimization of an incompletely specified function*. Instead, here we show that to explore all solutions we must set up the problem as the *minimization of a Boolean relation*, because there are don't care conditions that cannot be expressed by single cubes. In the experiments we report major improvements with respect to the previously published results.**

## I. Introduction

Logic optimization of digital circuits explores different realizations of a logic circuit to improve design parameters like area, speed, power consumption, etc. This goal is achieved exploiting both the flexibility allowed by the specification (external don't care conditions), and the internal or structural flexibility due to the specific structure of the implementation (e.g., satisfiability don't cares to model limited controllability, and observability don't cares to model limited observability).

An interesting case of structural flexibility arises when the outputs of some Boolean functions are combined by a disjunction, because if a single function produces the value 1 for some input combinations, we do not care about the output values produced by the remaining functions for such input.

In this context, decompositions with respect to specific critical variables are suitable to move the signals with the highest switching activity closer to the outputs (for low power consumption), or to move late-arriving signals closer to the outputs (to decrease worst-case delay). To favor area minimization while pushing signals ahead in the circuit, extended forms of Shannon cofactoring, *P-circuits*, were investigated in [1], [2], [3], where the expansion is with respect to an orthogonal basis $\overline{x}_i \oplus p$ (i.e., $x_i = p$), and $x_i \oplus p$ (i.e., $x_i \neq p$), where $p$ is a function defined over all variables except the critical variable $x_i$.

Let $f_{x_i=p}$ and $f_{x_i \neq p}$ be the projections of a function $f$ onto $x_i = p$ and $x_i \neq p$, and let $I = f_{x_i=p} \cap f_{x_i \neq p}$ be the points common to the two projections. A function $f$ can be decomposed into three Boolean functions combined by a disjunction: $f = (\overline{x}_i \oplus p)f^{=} + (x_i \oplus p)f^{\neq} + f^I$, where $f^{=} \subseteq f_{x_i=p}$, $f^{\neq} \subseteq f_{x_i \neq p}$, and $f^I \subseteq I$. The circuits synthesized according to this structure are called P-circuits when the blocks $f^{=}$, $f^{\neq}$, and $f^I$ are realized by sums-of-products. Shannon decomposition corresponds to the special case where $p = 0$, $f^{=} = f_{x_i=p}$, $f^{\neq} = f_{x_i \neq p}$, and $f^I = \emptyset$.

While the cofactoring variable $x_i$ is chosen among the most critical variables, the selection of a suitable $p$ for a given function is still an open and interesting problem.

In this paper, we address the problem of computing and using the complete flexibility of P-circuits, for which the previous disjunctive paradigm applies because, if a point $q$ of $f$ is covered in one set ($f^{=}$, $f^{\neq}$, or $f^I$), $q$ may be considered as a don't care in the other sets containing it.

The computation of the complete flexibility of P-circuits has been already described in the literature [1], [2], [3], but the algorithms proposed so far for its exploitation may fail to find the best implementation, because they cast the optimization problem as the *minimization of an incompletely specified function*. Instead, in this paper we show that to explore all solutions we must set up the problem as the *minimization of a Boolean relation*, because there are combinations of don't care conditions that cannot be expressed by single cubes. In this paper we then model the optimal P-circuit decomposition problem by means of Boolean relations, we implement the algorithm and report major improvements with respect to the previously published results. In fact, the P-circuits synthesized with Boolean relations are more compact then the corresponding P-circuits proposed in [3] in about 92% of our experiments, with an average gain in area of 33%.

The paper is organized as follows. Section II introduces P-Circuits. Section III describes how to minimize P-Circuits using Boolean relations. Experiments on a set of benchmarks are reported in Section IV, and Section V concludes the paper.

## II. P-circuits

A P-circuit is a network where the dependence on a given variable $x_i$ (e.g., the variable with more switching activity or with higher delay) is projected away from the rest of the circuit. In this section we first briefly review the P-circuits based on SOP minimization [1], [2], [3]. We then give a new formulation of the associated minimization problem, both for completely and incompletely specified functions, better suited to be formalized via Boolean relations.

### A. Completely Specified Functions

Let $f$ be a completely specified Boolean function depending on the set $\{x_1, \ldots, x_n\}$ of $n$ binary variables. The classical Shannon decomposition $f = \overline{x}_i f|_{\overline{x}_i} + x_i f|_{x_i}$, and the more general EXOR-based decomposition ([4], [5]) $f = (\overline{x}_i \oplus p) f|_{x_i=p} + (x_i \oplus p) f|_{x_i \neq p}$ (where $p$ is a function non-

978-3-9815370-0-0/DATE13/©2013 EDAA

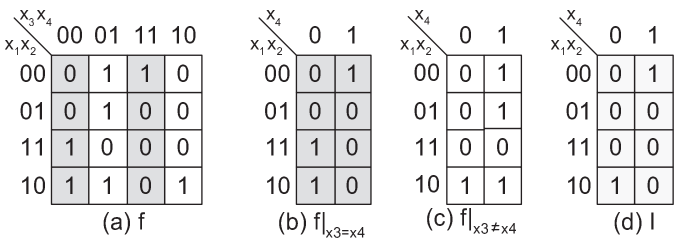

Fig. 1. A function $f$ (a), the corresponding projected functions $f|_{x_3=x_4}$ and $f|_{x_3 \neq x_4}$ (b) and (c), and their intersection I (d).

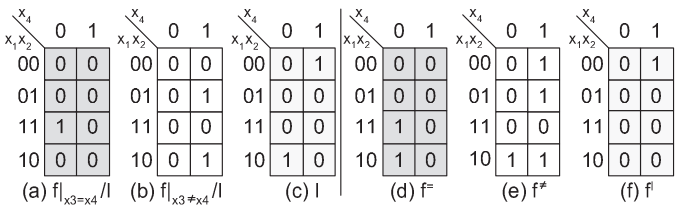

Fig. 2. More refined decompositions for the function in Fig. 1.

depending on $x_i$)[1] are suitable for keeping $x_i$ separated from the rest of the circuit, but are not oriented to area minimization. In fact, $f|_{x_i=p}$, and $f|_{x_i \neq p}$ do not depend on the variable $x_i$, but the cubes of $f$ intersecting both subsets $x_i = p$ and $x_i \neq p$ are split into two smaller subcubes when they are projected onto $f|_{x_i=p}$, and $f|_{x_i \neq p}$, respectively. For example, consider the function $f$ in Figure 1 (a), the variable $x_i = x_3$ and the simple function $p(x_1, x_2, x_4) = x_4$. The projection of $f$ w.r.t. $x_3 = x_4$ and $x_3 \neq x_4$ are depicted in Figures 1 (b) and (c), respectively. We note that the cube $\overline{x}_1\overline{x}_2x_4$ of $f$ is split into two separate minterms $\overline{x}_1\overline{x}_2x_4$ and $\overline{x}_1\overline{x}_2x_4$ in the two projected functions $f|_{x_3=x_4}$ and $f|_{x_3 \neq x_4}$ (Figures 1 (b), (c)).

To overcome this problem, in P-circuit synthesis, we keep unprojected some of the points of the original function. For this purpose, let $I = f|_{x_i=p} \cap f|_{x_i \neq p}$ be the intersection of the two cofactors $f|_{x_i=p}$ and $f|_{x_i \neq p}$. Note that the intersection $I$ contains the cubes whose products do not contain $x_i$ and that cross the two sets. Therefore, in order to overcome the splitting of these crossing cubes, we could keep $I$ unprojected, and project only the minterms in $f|_{x_i=p} \setminus I$ and $f|_{x_i \neq p} \setminus I$, obtaining the expression $f = (\overline{x}_i \oplus p)(f|_{x_i=p} \setminus I) + (x_i \oplus p)(f|_{x_i \neq p} \setminus I) + I$ . Note that $p$, $f|_{x_i=p} \setminus I$, $f|_{x_i \neq p} \setminus I$ and $I$ do not depend on $x_i$. Following the previous example, see Figures 2 (a), (b), and (c).

However, the points that are in $I$ could be exploited to form bigger cubes in the projected sets. For example, the point 001 of $I$ would be useful for forming the cube $\overline{x}_1x_4$ in $f|_{x_3 \neq x_4} \setminus I$, see Figure 2 (b), indeed such cube was originally in $f|_{x_3 \neq x_4}$, see Figure 1 (c). Therefore, if a point is in $I$ and is useful for a better minimization of the projected parts, it can be kept both in the projection and in the intersection (see, for example, the point 001 in Figures 2 (e) and (f)). Moreover, if a point is covered in both the projected sets it is not necessary to cover it in the intersection. In our example, the point 100 of $I$ is used for forming bigger cubes in both the projections (Figures 2 (d) and (e)), thus it can be removed from the intersection (see Figure 2 (f)).

In the following, we indicate a SOP circuit implementing a Boolean function $f$ with $S(f)$, and we denote optimal circuits (in the sense of two-level minimization) with a star, as $S^*(f)$.

From the previous observations, we can infer that the projected sub-circuits should cover at least $f|_{x_i=p} \setminus I$ and $f|_{x_i \neq p} \setminus I$, and must be contained in $f|_{x_i=p}$ and $f|_{x_i \neq p}$, respectively. Moreover, the part of the circuit that is not projected should be contained in the intersection $I$.

In summary, we rephrase the definition of a P-circuit given in [3] as follows.

*Definition 1:* A *P-circuit* of a completely specified function $f$ is the circuit $P(f)$ denoted by the expression:

$$P(f) = (\overline{x}_i \oplus S(p))\, S(f^{=}) + (x_i \oplus S(p))\, S(f^{\neq}) + S(f^I)$$

where

1) $(f|_{x_i=p} \setminus I) \subseteq f^{=} \subseteq f|_{x_i=p}$
2) $(f|_{x_i \neq p} \setminus I) \subseteq f^{\neq} \subseteq f|_{x_i \neq p}$
3) $\emptyset \subseteq f^I \subseteq I$
4) $P(f) = f$

Therefore, the synthesis idea of P-circuits is to construct a network for $f$ by appropriately choosing the sets $f^{=}$, $f^{\neq}$, and $f^I$ as building blocks. The cofactors and the intersection can be synthesized in any framework of logic minimization. Here we focus on the standard SOP synthesis. Thus, we represent $f^{=}$, $f^{\neq}$, and $f^I$ as sums of products.

For example, the expression $P(f) = (x_3 \oplus x_4)(x_1\overline{x}_2 + \overline{x}_1x_4) + (\overline{x}_3 \oplus x_4)(x_1\overline{x}_4) + \overline{x}_1\overline{x}_2x_4$ is a P-circuit for the function $f$ defined in Figure 1 (a). $P(f)$ is derived from the functions $f^{=}$, $f^{\neq}$, and $f^I$ in Figures 2 (d), (e), and (f).

An *optimal P-circuit* $P^*(f)$ for the function $f$ is a P-circuit with minimum cost that can be synthesized for $f$. The P-circuit described in the previous example is an optimal P-circuit w.r.t. the number of literals in the SOP forms.

## *B. Incompletely Specified Functions*

Consider now an incompletely specified Boolean function $f = \{f^{on}, f^{dc}\}$. For the sake of simplicity, suppose that $f^{on} \cap f^{dc} = \emptyset$; otherwise, following the usual semantics, we consider $f^{on} \setminus f^{dc}$ as the on-set of $f$. Let $I$ be the intersection of the projections of $f$ onto the two sets $x_i = p$ and $x_i \neq p$:

$$I = (f^{on}|_{x_i=p} \cup f^{dc}|_{x_i=p}) \cap (f^{on}|_{x_i \neq p} \cup f^{dc}|_{x_i \neq p}).$$

We generalize the definition of P-circuit in the following way.

*Definition 2:* A *P-circuit* of an incompletely specified function $f = \{f^{on}, f^{dc}\}$ is the circuit $P(f)$ denoted by the expression:

$$P(f) = (\overline{x}_i \oplus S(p))\, S(f^{=}) + (x_i \oplus S(p))\, S(f^{\neq}) + S(f^I)$$

where

1) $(f^{on}|_{x_i=p} \setminus I) \subseteq f^{=} \subseteq f^{on}|_{x_i=p} \cup f^{dc}|_{x_i=p}$
2) $(f^{on}|_{x_i \neq p} \setminus I) \subseteq f^{\neq} \subseteq f^{on}|_{x_i \neq p} \cup f^{dc}|_{x_i \neq p}$
3) $\emptyset \subseteq f^I \subseteq I$
4) $f^{on} \subseteq P(f) \subseteq f^{on} \cup f^{dc}$.

[1]Note that the Shannon decomposition is a particular EXOR-based decomposition where p=0.

## III. Minimization of P-circuits

The problem of minimizing a Boolean function in P-circuit form can be formulated in a very nice and clear way using Boolean relations, as explained in the following subsections, first for completely specified and then for incompletely specified functions.

### A. Boolean Relations

In this subsection we recall the theory of Boolean relations.

The concept of Boolean relations was introduced in [6] as a more general scheme for the incomplete specification of logic networks. In fact, the conditions under which a multi-output logic network can be simplified cannot always be completely represented using don't cares.

*Definition 3 ([6]):* A *Boolean relation* is a one-to-many multi-output Boolean mapping $\mathcal{R} : \mathbb{B}^n \to \mathbb{B}^m$, where $\mathbb{B} = \{0, 1\}$. $\mathbb{B}^n$ and $\mathbb{B}^m$ are called the *input* and *output sets* of $\mathcal{R}$.

A Boolean relation $\mathcal{R}$ can be considered a generalization of a Boolean function, where a point in the input set $\mathbb{B}^n$ can be associated with several points in the output set $\mathbb{B}^m$; indeed, because of the one-to-many nature of Boolean relations, there may be several equivalent outputs for a given input. Also, notice that $\mathcal{R}$ is a subset of the Cartesian product $\mathbb{B}^n \times \mathbb{B}^m$.

A relation $\mathcal{R}$ is *well-defined* if for all $x \in \mathbb{B}^n$, there is $y \in \mathbb{B}^m$ such that $(x, y) \in \mathcal{R}$. A well defined Boolean relation is *functional* if every input $x$ is associated with one and only one output $y$.

To any relation $\mathcal{R}$ we can associate a set of *compatible* multi-output Boolean functions, i.e. the set of all functions $f$ such that, for all inputs $x \in \mathbb{B}^n$, $f(x)$ is contained in the set $\mathcal{R}(x)$ of the outputs related to $x$. In this case, we write $f \subseteq \mathcal{R}$.

*Definition 4 ([7]):* The set of multi-output Boolean functions compatible with a Boolean relation $\mathcal{R} : \mathbb{B}^n \to \mathbb{B}^m$, is defined as $\mathcal{F}(\mathcal{R}) = \{f : \mathbb{B}^n \to \mathbb{B}^m \mid f \subseteq \mathcal{R} \text{ and } f \text{ is a function}\}$.

The problem of the optimal implementation of a Boolean relation $\mathcal{R}$ is the one of selecting, among the possible functions compatible with $\mathcal{R}$, one of minimum cost according to a given metric. More precisely

*Definition 5 ([7]):* The *solution* of a Boolean relation $\mathcal{R}$ is a multi-output Boolean function $f \in \mathcal{F}(\mathcal{R})$. The function $f$ is an *optimal solution* of $\mathcal{R}$ according to a given cost function $c$, if for all $f' \in \mathcal{F}(\mathcal{R})$, $c(f) \leq c(f')$.

Several exact and heuristic algorithms have been proposed for solving Boolean relations (see [7] for an overview of these methods, and for bibliographic references). For our particular minimization problem, we used the recursive algorithm proposed in [7] because of its ability in finding better solutions in shorter runtime than the previously known methods. In fact, this recursive solver is very efficient and able to explore a wide space of solution. Moreover it can be used both in an exact and heuristic mode and its cost function can be tuned for different parameters related to area or delay, when minimizing the relation.

### B. Completely Specified Functions

Let $f$ be a completely specified Boolean function depending on $n$ variables, $x_i$ one input variable, and $p$ a function depending on all input variables of $f$ but $x_i$. Consider the two cofactors $f|_{x_i=p}$ and $f|_{x_i \neq p}$, obtained by projecting $f$ onto the sets $x_i = p$ and $x_i \neq p$, and their intersection $I = f|_{x_i=p} \cap f|_{x_i \neq p}$.

As recalled in Section II, in order to minimize $f$ in P-circuit form, we must find a refinement of the starting partition of the minterms of $f$ into the three disjoint sets $f|_{x_i=p} \setminus I$, $f|_{x_i \neq p} \setminus I$ and $I$. This refinement, which leads to the three new sets $f^=$, $f^{\neq}$, and $f^I$, consists in *(i)* adding back some points of $I$ to $f|_{x_i=p} \setminus I$ or $f|_{x_i \neq p} \setminus I$, precisely the points that could be used to obtain bigger cubes in their SOP representation, and *(ii)* eventually subtracting from $I$ some of the points in the intersection of the refined cofactors, in order to find a more compact SOP representation for the intersection set. The final P-circuit for $f$ is then given by three minimal SOPs representing $f^=$, $f^{\neq}$, and $f^I$, opportunely combined with a minimal SOP for $p$.

So, the problem to solve is that of finding the sets $(f^=, f^{\neq}, f^I)$ that lead to a P-circuit of minimal cost, according to a given cost metric. We now show how this problem can be nicely formalized and efficiently solved using Boolean relations. Our aim is to define a relation $\mathcal{R}$ whose set of compatible functions $\mathcal{F}(\mathcal{R})$ have implementations corresponding exactly to the set of all possible P-circuits for $f$, so that an optimal solution of $\mathcal{R}$ is an optimal P-circuit $P^*(f)$ for $f$.

Let $\mathcal{R}_f : \mathbb{B}^{n-1} \to \mathbb{B}^3$ be a Boolean relation, whose input set is the space spanned by the variables $x_1 \dots x_{i-1} x_{i+1} \dots x_n$, while the output set describes all possible tuples of functions $f^=$, $f^{\neq}$, $f^I$ defining a P-circuit for $f$. From Definition 1, we can observe that the sets $f|_{x_i=p} \setminus I$ and $f|_{x_i \neq p} \setminus I$ must be always contained in $f^=$ and $f^{\neq}$, respectively. Thus, all points in $f|_{x_i=p} \setminus I$ are associated to the output 100, and all points in $f|_{x_i \neq p} \setminus I$ are associated to the output 010 of $\mathcal{R}_f$. Moreover, we can also note that the possible P-circuit implementations of $f$ differ in the distribution of the points of $I$ among the three sets $f^=$, $f^{\neq}$, $f^I$. In particular, each minterm in $I$ can be associated to one of the following outputs of $\mathcal{R}_f$:

- 001 if it belongs only to $f^I$, that is, it has been kept in $I$, and not added to $f^=$ and $f^{\neq}$;
- 101 if it has been added to $f^=$ and it is kept in $I$;
- 011 if it has been added to $f^{\neq}$ and it is kept in $I$;
- 111 if it has been added to $f^=$ and $f^{\neq}$, and it is kept in $I$;
- 110 if it has been added to $f^=$ and $f^{\neq}$, and it is removed from $I$.

The possible outputs corresponding to the points in $I$ can be represented in a compact way using don't cares: $--1$ and $11-$. Indeed, if a point is covered in the intersection set, then it becomes a don't care point in the two projection sets $x_i = p$ and $x_i \neq p$, while if it is covered in both projection sets, it becomes a don't care for $I$. Since all points of $I$ must be covered either in $f^I$, or both in $f^=$ and $f^{\neq}$, all other possible output choices for $I$ would not define a P-circuit for $f$. The relation $\mathcal{R}_f$ can therefore be defined as follows:

| $x_1 \dots x_{i-1} x_{i+1} \dots x_n$ | $\mathcal{R}_f$ |
|---|---|
| points in $f\|_{x_i=p} \setminus I$ | 100 |
| points in $f\|_{x_i \neq p} \setminus I$ | 010 |
| points in $I$ | $\{--1, 11-\}$ |
| all other points | 000 |

Observe that $\mathcal{R}_f$ is truly a relation, as it is not functional on the points of $I$.

With this formalism, we can rephrase our minimization problem as the problem of finding an optimal implementation of $\mathcal{R}_f$, that is, of selecting among all possible three-output functions compatible with $\mathcal{R}_f$, each corresponding to a tuple $f^=$, $f^{\neq}$, and $f^I$, the one whose overall SOP representation is minimal.

For example, consider the function in Figure 1, and its projections over the sets $x_3 = x_4$ and $x_3 \neq x_4$. The intersection between the two cofactors $f|_{x_3=x_4} = \{001, 110, 100\}$ and $f|_{x_3 \neq x_4} = \{001, 011, 100, 101\}$ contains two points $\{001, 100\}$. Thus we have $I = \{001, 100\}$, $f|_{x_3=x_4} \setminus I = \{110\}$, $f|_{x_3 \neq x_4} \setminus I = \{011, 101\}$. The corresponding relation is

| $x_1 x_2 x_4$ | $\mathcal{R}_f$ |
|---|---|
| 001 | $\{11-, --1\}$ |
| 011 | 010 |
| 100 | $\{11-, --1\}$ |
| 101 | 010 |
| 110 | 100 |

where all missing points are associated to the output 000. It can be verified that among all functions compatible with $\mathcal{R}_f$, the one whose overall SOP representation is minimal (w.r.t. the number of literals) is

| $x_1 x_2 x_4$ | $(f^=, f^{\neq}, f^I)$ |
|---|---|
| 001 | 0 1 1 |
| 011 | 0 1 0 |
| 100 | 1 1 0 |
| 101 | 0 1 0 |
| 110 | 1 0 0 |

that defines precisely the sets $f^=$, $f^{\neq}$ and $f^I$ shown in Figures 2 (d), (e), and (f).

## C. Incompletely Specified Functions

Let us now consider the P-circuit minimization of incompletely specified functions. The construction of $\mathcal{R}_f$ must consider more cases. In fact, the possible P-circuit implementations depend not only on the distribution of the points of $I$ among the three sets $f^=$, $f^{\neq}$, $f^I$, but also in the handling of the don't care points of the two cofactors, as we can use them to build bigger cubes in the two projection spaces $x_i = p$ and $x_i \neq p$, and possibly in the intersection $I$.

To correctly define the relation $\mathcal{R}_f$ we must partition the input set $\mathbb{B}^{n-1}$ into nine subsets, each corresponding to a combination of the values $\{0, 1, -\}$ assumed by the two cofactors $f|_{x_i=p}$ and $f|_{x_i \neq p}$, as shown in Figure 3. Given $x \in \mathbb{B}^{n-1}$, we have the following cases:

| $f\|_{xi=p}$ \ $f\|_{xi\neq p}$ | 0 | 1 | - |
|---|---|---|---|
| 0 | 000 | 010 | 0-0 |
| 1 | 100 | --1<br>11- | --1<br>1-- |
| - | -00 | --1<br>-1- | --- |

Fig. 3. Outputs for the relation $\mathcal{R}_f$ when $f$ is an incompletely specified function.

1) $[f|_{x_i=p}(x) = 0,\ f|_{x_i \neq p}(x) = 0]$
$x$ is a point of both off-sets, thus $\mathcal{R}_f(x) = 000$.
2) $[f|_{x_i=p}(x) = 0,\ f|_{x_i \neq p}(x) = 1]$
Definition 2 immediately implies that $\mathcal{R}_f(x) = 010$.
3) $[f|_{x_i=p}(x) = 1,\ f|_{x_i \neq p}(x) = 0]$
As before, Definition 2 implies that $\mathcal{R}_f(x) = 100$.
4) $[f|_{x_i=p}(x) = -,\ f|_{x_i \neq p}(x) = 0]$
The don't cares of $f|_{x_i=p}$, corresponding to zeros of the other cofactor, can be exploited to get a smaller SOP form for $f^=$, thus we pose $\mathcal{R}_f(x) = -00$.
5) $[f|_{x_i=p}(x) = 0,\ f|_{x_i \neq p}(x) = -]$
As before, the don't cares of $f|_{x_i \neq p}$, corresponding to zeros of $f|_{x_i=p}$, can be exploited to get a smaller SOP form for $f^{\neq}$, thus we pose $\mathcal{R}_f(x) = 0-0$.
6) $[f|_{x_i=p}(x) = 1,\ f|_{x_i \neq p}(x) = 1]$
The points in the intersection between the on-sets of the two cofactors can be treated exactly as done for completely specified functions, so we have $\mathcal{R}_f(x) = \{--1, 11-\}$.
7) $[f|_{x_i=p}(x) = 1,\ f|_{x_i \neq p}(x) = -]$
For each point $x$ in the on-set of $f|_{x_i=p}$ and in the don't care set of $f|_{x_i \neq p}$ we have two choices: either it is covered in the projection set $x_i = p$, thus becoming a don't care point in both the intersection and the other projection set $x_i \neq p$, or it can be covered in the intersection and be a don't care in both projection sets. Thus, we pose $\mathcal{R}_f(x) = \{1--, --1\}$.
8) $[f|_{x_i=p}(x) = -,\ f|_{x_i \neq p}(x) = 1]$
Analogously, for each point $x$ in don't care set of $f|_{x_i=p}$ and in the on-set of $f|_{x_i \neq p}$, we pose $\mathcal{R}_f(x) = \{-1-, --1\}$.
9) $[f|_{x_i=p}(x) = -,\ f|_{x_i \neq p}(x) = -]$
Finally, don't cares of both cofactors can be used as don't cares in all the sets, i.e., $\mathcal{R}_f(x) = ---$.

As for completely specified functions, we can rephrase the synthesis of an incompletely specified function $f$ in P-circuit form as the problem of finding an optimal implementation of $\mathcal{R}_f$, i.e., of selecting among all possible three-output functions compatible with $\mathcal{R}_f$, the one defining a tuple $f^=$, $f^{\neq}$, and $f^I$ whose overall SOP representation is minimal.

*D. Optimality Issues*

We now show that the set $\mathcal{F}(\mathcal{R}_f)$ of all three-output functions compatible with the relation $\mathcal{R}_f$ specifies exactly the set of all tuples $f^=$, $f^{\neq}$, and $f^I$ defining a P-circuit for a Boolean function $f$, so that an optimal solution of $\mathcal{R}_f$ is an optimal P-circuit $P^*(f)$ for $f$. We consider only the problem of minimizing an incompletely specified function in P-circuit form, since it subsumes the problem of minimizing a completely specified function, whose don't care set is just the empty set. Due to space limitations, formal proofs will be discussed in the extended version of this paper.

Let $f$ be an incompletely specified Boolean function, $x_i$ an input variable, and $R_f$ the Boolean relation constructed as explained before. For simplicity, we only consider the case $p = 0$, where $p$ does not need any minimization (note that the experimental results in [1] show that the best choice for the projection function $p$ is the simplest $p = 0$). Recall that the three output variables of $R_f$ are used to describe the tuple of functions $f^=$, $f^{\neq}$, and $f^I$ defining a P-circuit for $f$: the first two outputs define $f^=$ and $f^{\neq}$, and the third defines $f^I$. Thus, each function in $\mathcal{F}(\mathcal{R}_f)$ corresponds to a possible tuple.

*Theorem 1:* The set $\mathcal{F}(\mathcal{R}_f)$ of three-output functions compatible with the relation $\mathcal{R}_f$ defines exactly the set of tuples $f^=$, $f^{\neq}$, and $f^I$ occurring in all possible P-circuit implementations of $f$.

We finally state that under the hypothesis that the projection function $p$ is the constant function $p = 0$, an optimal solution of $\mathcal{R}_f$ provides an optimal P-circuit for $f$.

*Corollary 1:* For $p = 0$, the optimal solution of the Boolean relation $\mathcal{R}_f$, according to a given cost function $\mu$ chosen to evaluate the circuits $P(f)$, defines an optimal P-circuit $P^*(f)$ for the function $f$ w.r.t. the same cost function $\mu$.

The general case, where we project w.r.t. any given function $p$, can be studied in a similar way. This issue will be discussed in an extended version of this paper. Here we only anticipate that the problem must be reformulated with a four output relation whose output set describes all the possible tuples of functions $f^=$, $f^{\neq}$, $f^I$, and $p$.

## IV. Experimental Results

In this section we report the experimental results for the P-circuits derived by the synthesis of Boolean relations. Due to space limitations and since the experimental results in [1], [2] show that the best choice for the projection function $p$ is often $p = 0$, we evaluate and report the area, delay and power dissipation just for $p = 0$, using $x_i = x_1$. The algorithms have been implemented in C, using the CUDD library for OBDDs to represent Boolean functions, and BREL [7] for the synthesis of Boolean relations since, as far as we know, it finds better solutions in shorter runtime than the previously known methods. The experiments were run on a Linux Intel Core i7, 3.40 GHz CPU with 8 GB of main memory. The benchmarks are taken from LGSynth93 [8]. Multioutput benchmarks were synthesized by minimizing each single output independently from the others. We report in the following a significant subset of the functions as representative indicators of our experiments.

To evaluate the obtained circuits, we ran them using the SIS system with the MCNC library for technology mapping and the SIS command `map -W -f 3 -s`. To show the gain in area of P-circuits derived using Boolean relations, we compare our results with the algorithm based on don't cares reported in [3]. To highlight the importance of projecting the intersection, we report also the results regarding circuits decomposed with standard Shannon decomposition (therefore without any intersection), called here *S-circuits*. We compare our results also with plain SOP forms, whose minimization does not even take into account the critical variable $x_i$. The SOP circuits and the projections of the SOP components of the S-circuits have been synthesized using ESPRESSO in the heuristic mode.

In Table I we compare synthesis time (in seconds), mapped area and delay of P-circuits, S-circuits, and SOP forms for a significant subset of the benchmarks. The first column reports the name of the benchmarks and the number of their inputs and outputs. The following ones report, by groups of three, the synthesis times in seconds, and the areas and delays estimated by SIS. The first two groups, labeled P-circuit $\mu_L$ and P-circuit $\mu_{BDD}$, refer to P-circuits synthesized with the new algorithm based on Boolean relations with cost function $\mu_L$ that minimizes the number of literals, and $\mu_{BDD}$ that minimizes the size of the BDDs used to represent the relations. The next group refers to P-circuits synthesized with the minimization strategy proposed in [3]. The last two groups provide the results for S-circuits and SOP forms. For each benchmark we underline in bold the circuit that exhibits better area results.

The results show that modeling the P-circuit minimization problem using Boolean relations pays significantly. In fact, P-circuits synthesized with Boolean relations (P-circuit $\mu_L$ and P-circuit $\mu_{BDD}$) turned out to be more compact than the corresponding P-circuits proposed in [3] in about 92% and 78% of our experiments, respectively. The area gain of P-circuits synthesized with Boolean relations and cost function $\mu_L$ ($\mu_{BDD}$) is 33% (25%) in the average w.r.t. P-circuits in [3]. Several benchmarks have gain above 60% (see, for example *m4* with 66% of gain for $\mu_L$ and 63% for $\mu_{BDD}$). The area gain w.r.t. SOP forms is also very interesting, since P-circuits are designed for dealing with critical variables and thus area is of secondary importance. The P-circuits synthesized with Boolean relations and cost function $\mu_L$ ($\mu_{BDD}$) are more compact then the corresponding SOP forms in about 72% (60%) of our experiments, with an average gain in area of 26% (18%).

Comparing the performances of the two new algorithms to the P-circuit algorithm proposed in [3], we can notice how the cost function can be critical: $\mu_L$ can be quite time-expensive (4214% penalty in computational time, on average, w.r.t. [3]), while $\mu_{BDD}$ is the best-performing algorithm (56% gain in computational time, on average, w.r.t. [3]). For a complete comparison of average gains see Table II. We can observe that the algorithm based on Boolean relations with cost function $\mu_{BDD}$ exhibits the best trade-off between area minimization and computational time.

To evaluate power dissipation, the circuits were synthesized using the following equal-delay gates in a 180 nm CMOS technology: inverter, 2,3,4-input NAND, 2,3,4-input NOR and 2-input XOR gates. Since in CMOS technology at 180 nm the switching power is the predominant component of the

TABLE I. SYNTHESIS TIME (IN SECONDS), MAPPED AREA AND DELAY OF P-CIRCUITS, S-CIRCUITS, AND SOP FORMS.

| **Benchmark** | **P-circuit $\mu_L$** | | | **P-circuit $\mu_{BDD}$** | | | **P-circuit [3]** | | | **S-circuit** | | | **SOP** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (in/out) | **Time** | **Area** | **Delay** | **Time** | **Area** | **Delay** | **Time** | **Area** | **Delay** | **Time** | **Area** | **Delay** | **Time** | **Area** | **Delay** |
| **b10** (15/11) | 9.75 | **850** | 39.1 | 0.18 | 938 | 38.7 | 0.05 | 1067 | 42.0 | 0.07 | 1113 | 47.5 | 0.01 | 881 | 45.2 |
| **b2** (16/17) | 103.82 | **3591** | 68.2 | 1.68 | 3757 | 69.9 | 0.66 | 4479 | 82.8 | 10.19 | 6281 | 115.4 | 0.01 | 3957 | 73.0 |
| **b3** (32/20) | 168.10 | **949** | 28.4 | 0.13 | 1002 | 29.5 | 0.93 | 1134 | 39.0 | 0.82 | 1284 | 40.1 | 0.03 | 1095 | 44.7 |
| **ex1010** (10/10) | 50.65 | **2923** | 81.0 | 0.61 | 4170 | 96.1 | 0.39 | 4271 | 102.3 | 0.58 | 4818 | 113.5 | 0.31 | 4254 | 95.1 |
| **ex4** (128/28) | 18.51 | **800** | 21.8 | 1.15 | 1099 | 27.3 | 0.22 | 806 | 25.0 | 0.40 | 1520 | 29.6 | 0.10 | 801 | 25.0 |
| **exam** (10/10) | 5.78 | **408** | 30.0 | 0.14 | 475 | 28.8 | 0.08 | 655 | 29.0 | 0.12 | 617 | 32.7 | 0.07 | 526 | 31.8 |
| **exep** (30/63) | 1.89 | 1305 | 30.5 | 0.03 | 1315 | 28.5 | 0.04 | 1347 | 31.6 | 0.07 | 1321 | 30.5 | 0.01 | **1275** | 36.1 |
| **gary** (15/11) | 11.98 | **1000** | 41.1 | 0.19 | 1096 | 44.9 | 0.04 | 1189 | 44.9 | 0.04 | 1304 | 50.3 | 0.01 | 1030 | 53.9 |
| **ibm** (48/17) | 13.33 | 741 | 30.1 | 0.58 | 1117 | 35.4 | 0.04 | 741 | 30.1 | 0.08 | 1368 | 42.9 | 0.01 | **700** | 29.0 |
| **in3** (35/29) | 4.32 | **929** | 34.5 | 0.15 | 968 | 36.2 | 0.06 | 1157 | 35.8 | 0.04 | 1153 | 38.6 | 0.01 | 1111 | 34.1 |
| **in4** (32/20) | 167.37 | **1002** | 28.5 | 0.13 | 1055 | 29.6 | 0.77 | 1192 | 38.0 | 0.56 | 1331 | 40.2 | 0.02 | 1077 | 44.8 |
| **jbp** (36/57) | 4.42 | **1059** | 33.8 | 0.16 | 1310 | 35.4 | 0.05 | 1193 | 38.6 | 1.38 | 1690 | 40.0 | 0.02 | 1115 | 35.9 |
| **m4** (8/16) | 7.20 | **783** | 36.6 | 0.01 | 849 | 39.5 | 0.02 | 2274 | 71.4 | 0.02 | 2766 | 85.3 | 0.03 | 1778 | 54.2 |
| **mainpla** (27/54) | 338.55 | **15118** | 205.1 | 6.58 | 15610 | 217.5 | 12.17 | 26531 | 345.9 | 337.04 | 24421 | 335.2 | 0.08 | 25529 | 371.0 |
| **max1024** (10/6) | 33.96 | **1408** | 47.3 | 0.02 | 1554 | 58.4 | 0.08 | 2755 | 70.6 | 0.07 | 2534 | 75.2 | 0.14 | 1690 | 53.7 |
| **misg** (56/23) | 1.38 | 153 | 18.2 | 0.04 | 167 | 18.2 | 0.01 | 153 | 18.2 | 0.04 | 348 | 23.0 | 0.01 | **152** | 18.2 |
| **mish** (94/43) | 0.54 | **168** | 10.0 | 0.07 | 207 | 10.0 | 0.09 | **168** | 10.0 | 0.29 | 427 | 13.9 | 0.01 | **168** | 9.4 |
| **misj** (35/14) | 0.52 | **81** | 9.7 | 0.03 | 125 | 9.1 | 0.01 | **81** | 8.9 | 0.01 | 184 | 13.9 | 0.01 | **81** | 8.9 |
| **pdc** (16/40) | 1.83 | **683** | 26.8 | 0.03 | 722 | 26.4 | 0.40 | 1555 | 46.8 | 0.53 | 826 | 34.7 | 0.44 | 1633 | 44.0 |
| **prom2** (9/21) | 18.61 | **4612** | 115.1 | 0.01 | 4905 | 117.0 | 0.42 | 7462 | 171.4 | 0.24 | 7226 | 175.1 | 0.18 | 6775 | 185.1 |
| **spla** (16/46) | 3.24 | **1826** | 50.6 | 0.01 | 1935 | 50.7 | 0.17 | 2284 | 59.9 | 0.15 | 2239 | 53.8 | 0.21 | 2470 | 68.6 |
| **ts10** (22/16) | 43.40 | 942 | 36.5 | 0.50 | 1410 | 44.9 | 0.17 | 942 | 36.5 | 0.16 | 1806 | 66.1 | 0.01 | **901** | 35.2 |
| **vg2** (25/8) | 5.15 | 577 | 22.8 | 0.06 | 632 | 23.9 | 0.01 | 603 | 23.2 | 0.01 | 546 | 25.6 | 0.01 | **341** | 18.6 |
| **x6dn** (39/5) | 10.64 | **760** | 30.8 | 0.26 | 788 | 31.6 | 0.01 | 874 | 35.9 | 0.01 | 885 | 35.1 | 0.01 | 762 | 31.2 |
| **x7dn** (66/15) | 64.15 | 2378 | 51.1 | 0.01 | 2478 | 51.8 | 0.51 | 2501 | 63.8 | 0.54 | 2371 | 49.9 | 0.12 | **1544** | 46.1 |
| **x9dn** (27/7) | 3.30 | 412 | 24.1 | 0.04 | 425 | 24.5 | 0.02 | 412 | 24.1 | 0.02 | 530 | 27.9 | 0.01 | **384** | 23.0 |
| **xparc** (41/73) | 84.08 | **9175** | 121.2 | 0.32 | 9280 | 126.9 | 0.51 | 13478 | 187.3 | 0.53 | 13357 | 180.4 | 0.14 | 12678 | 168.8 |

TABLE II. AVERAGE GAIN OF P-CIRCUITS BASED ON BOOLEAN RELATIONS

| | **P-circuit $\mu_L$** | | | **P-circuit $\mu_{BDD}$** | | |
|---|---|---|---|---|---|---|
| **Average gain** | **Time** | **Area** | **Delay** | **Time** | **Area** | **Delay** |
| **w.r.t. S-circuit** | -383% | 37% | 29% | 95% | 30% | 25% |
| **w.r.t. P-circuit** [3] | -4214% | 33% | 24% | 56% | 25% | 20% |
| **w.r.t. SOP** | -39412% | 26% | 19% | -304% | 18% | 14% |

overall power consumption, our model takes into account only this component. The circuits were analyzed with two different average values of input transition rates, giving a high-density switching activity for just one known signal (in our case $x_1$) and a low-density activity for all other signals. The number of input logic transitions for the high-density switching activity signal is approximately one hundred times larger than the number of input logic transitions for the low-density switching activity signals. The circuits have been simulated at transistor level using SPECTRE on a 1600 MHz Pentium 4 workstation.

TABLE III. COMPARISON OF POWER DISSIPATION

| | **P-circuit $\mu_L$** | **P-circuit $\mu_{BDD}$** |
|---|---|---|
| **w.r.t. S-circuit** | 65% | 61% |
| **w.r.t. P-circuit** [3] | 13% | 13% |
| **w.r.t. SOP** | 44% | 62% |

Table III summarizes the results obtained using Boolean relations. The power consumption of P-circuits synthesized with Boolean relations (P-circuit $\mu_L$ and P-circuit $\mu_{BDD}$) is lower than the power consumption of the corresponding P-circuits proposed in [3] in about 13% of our experiments. This percentage goes up if we compare P-circuits synthesized with Boolean relations against SOPs and S-circuits.

In summary, from Tables II and III, we observe that the P-circuits synthesized with the algorithm based on Boolean relations with cost function $\mu_{BDD}$ are a good trade-off between power dissipation, area/delay, and computational time.

## V. CONCLUSION AND FUTURE WORK

We addressed the problem of exploiting the complete flexibility of P-circuits, showing that to explore all possible solutions one must cast the problem as one of minimizing a Boolean relation. In the experiments we report major improvements with respect to the previously published results. Future work includes investigating the impact of using more general cofactoring $p$ functions, and addressing simultaneously multi-ouput functions trading-off quality of results vs. scalability.

## ACKNOWLEDGEMENT

We are grateful to D. Bañeres and J. Cortadella, for providing us with BREL tool [7] and for their technical advice.

## REFERENCES

[1] A. Bernasconi, V. Ciriani, G. Trucco, and T. Villa, "On Decomposing Boolean Functions via Extended Cofactoring," in *Design Automation and Test in Europe (DATE)*, 2009, pp. 1464–1469.

[2] ——, "Logic Synthesis by Signal-Driven Decomposition," in *Advanced Techniques in Logic Synthesis, Optimizations and Applications*, K. Gulati, Ed. Springer New York, 2011, pp. 9–29.

[3] A. Bernasconi, V. Ciriani, V. Liberali, G. Trucco, and T. Villa, "Synthesis of P-circuits for logic restructuring," *Integration*, vol. 45, no. 3, pp. 282–293, 2012.

[4] J. C. Bioch, "The complexity of modular decomposition of Boolean functions," *Discrete Applied Mathematics*, vol. 149, no. 1-3, pp. 1–13, 2005.

[5] P. Kerntopf, "New Generalizations of Shannon Decomposition," in *Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, 2001, pp. 109–118.

[6] R. Brayton and F. Somenzi, "Boolean relations and the incomplete specification of logic networks," in *International Conference on Very Large Scale Integration - VLSI*, August 1989.

[7] D. Bañeres, J. Cortadella, and M. Kishinevsky, "A Recursive Paradigm to Solve Boolean Relations," *IEEE Transactions on Computers*, vol. 58, no. 4, pp. 512–527, 2009.

[8] S. Yang, "Logic synthesis and optimization benchmarks user guide version 3.0," Microelectronic Center, User Guide, 1991.