ΥΠΟΥΡΓΕΙΟ ΕΘΝΙΚΗΣ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ ΕΠΕΑΕΚ
ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
ΣΥΓΧΡΗΜΑΤΟΔΟΤΗΣΗ
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Η ΠΑΙΔΕΙΑ ΣΤΗΝ ΚΟΡΥΦΗ
Επιχειρησιακό Πρόγραμμα
Εκπαίδευσης και Αρχικής
Επαγγελματικής Κατάρτισης
ΕΛΛΑΔΑ 2008
Ανάπτυξη παντού. Ανάπτυξη για όλους.

Cell-Library Binding

Εισαγωγή

- ✓ **Library Binding (technology mapping)**: η μετατροπή ενός λογικού δικτύου σε διασύνδεση στοιχείων μίας βιβλιοθήκης (τεχνολογίας).
- ✓ Παρέχει ολοκληρωμένη κατασκευαστική αναπαράσταση του δικτύου.
- ✓ Δίνει την δυνατότητα της αντιστοίχισης του σχεδιασμού σε διαφορετικές τεχνολογίες και στυλ υλοποίησης.
- ✓ Η βιβλιοθήκη περιλαμβάνει βασικά στοιχεία (primitives). Το binding επιλέγει τα πιο κατάλληλα για μία υλοποίηση.
- ✓ Οι βασικές προσεγγίσεις είναι δύο: ευριστικές και κανόνες.
- ✓ Μεγαλύτερο ενδιαφέρον έχουν τα συνδυαστικά κυκλώματα αφού οι καταχωρητές έχουν εύκολη υλοποίηση.
- ✓ Η υλοποίηση λογικών εκφράσεων δύο επιπέδων γίνεται με την αποσύνθεσή τους σε δίκτυο πολλαπλών επιπέδων.

Πρόβλημα & Ανάλυση

- ✓ Κάθε στοιχείο μίας βιβλιοθήκης χαρακτηρίζεται από:

  α. Μία συνδυαστική λογική συνάρτηση μίας εξόδου.

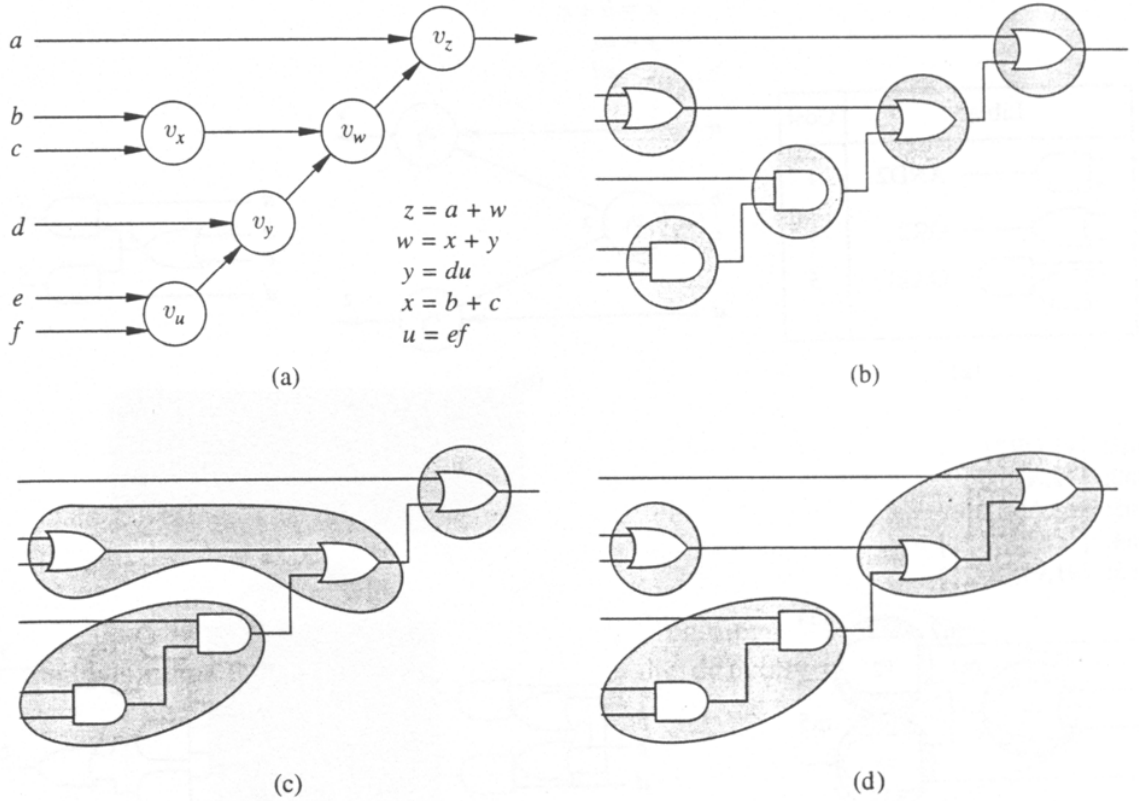
  β. Κόστος επιφάνειας

  γ. Καθυστερήσεις εισόδου/εξόδου

- ✓ Η βιβλιοθήκη περιέχει τις απαραίτητες πληροφορίες για την υλοποίηση κάθε στοιχείου (πχ physical layout)
- ✓ Η υλοποίηση βιβλιοθήκης στην περίπτωση των FPGAs διαφέρει.
- ✓ Στόχος του library binding είναι η ελαχιστοποίηση της επιφάνειας (υπό περιορισμούς ταχύτητας) ή της μέγιστης καθυστέρησης (υπό περιορισμούς επιφάνειας).
- ✓ Το πρόβλημα είναι υπολογιστικά δύσκολο (ακόμη και η ταυτοποίηση της ισοδυναμίας δύο δικτύων είναι δύσκολο πρόβλημα).

Πρόβλημα & Ανάλυση

- ✓ **Κάλυψη δικτύου από library cells**: αναγνώριση ενός υποδικτύου και αντικατάστασή του από στοιχεία της βιβλιοθήκης που βελτιστοποιούν κάποιο χαρακτηριστικό του.



Πρόβλημα & Ανάλυση

- ✓ Ένα κύτταρο ταιριάζει με ένα υποδίκτυο όταν είναι λειτουργικά ισοδύναμα. (Πιθανώς να έχουν και διαφορετικό αριθμό εισόδων).

Τετριμμένο binding:

- ➢ Αντιστοιχούμε κάθε κορυφή σε ένα κύτταρο της βιβλιοθήκης.
- ➢ Δεν είναι βέλτιστο ακόμη και αν το λογικό δίκτυο είναι βέλτιστο (δεν λαμβάνει υπόψη τα χαρακτηριστικά της τεχνολογίας).

- ✓ Στο πρόβλημα κάλυψης μελετάμε κάθε υποδίκτυο το οποίο έχει ρίζα μία κορυφή, και το αντιστοιχούμε σε ένα υποσύνολο κυττάρων.

Πρόβλημα & Ανάλυση

- ✓ Μία αναγκαία συνθήκη για να έχει λύση το πρόβλημα της κάλυψης είναι να υπάρχει τουλάχιστον μία αντιστοίχιση για κάθε κορυφή.
- ✓ Αυτή η συνθήκη ικανοποιείται συνήθως με decomposition.
- ✓ Σε κάθε αντιστοίχιση πρέπει να εξασφαλίζεται ότι οι είσοδες άλλων κορυφών συνεχίζουν να υπάρχουν.
- ✓ Ο αλγόριθμος Branch & Bound μπορεί να χρησιμοποιηθεί για την κάλυψη μικρών λογικών δικτύων.

Αλγόριθμος Branch & Bound

Πρόβλημα: Έχουμε ένα ZOLP με n μεταβλητές απόφασης $x=[x_1, x_2, \ldots, x_n]$.

Εξαντλητική Λύση: Δοκιμάζω όλες τις 2^n δυνατές τιμές.

Συστηματική Λύση: Επιλέγουμε μία μεταβλητή και

α) την θέτουμε στο 1 και λύνουμε το υπο-πρόβλημα στο οποίο έχουμε σβήσει την μεταβλητή και

β) την θέτουμε στο 0 και ξανα-λύνουμε το υποπρόβλημα

=

Δέντρο απόφασης με φύλλα όλες τις πιθανές λύσεις

(η επίσκεψη όλων των φύλων-λύσεων είναι εκθετική λύση στη μέση και χειρότερη περίπτωση).

Αλγόριθμος Branch & Bound

Branch & Bound Λύση:

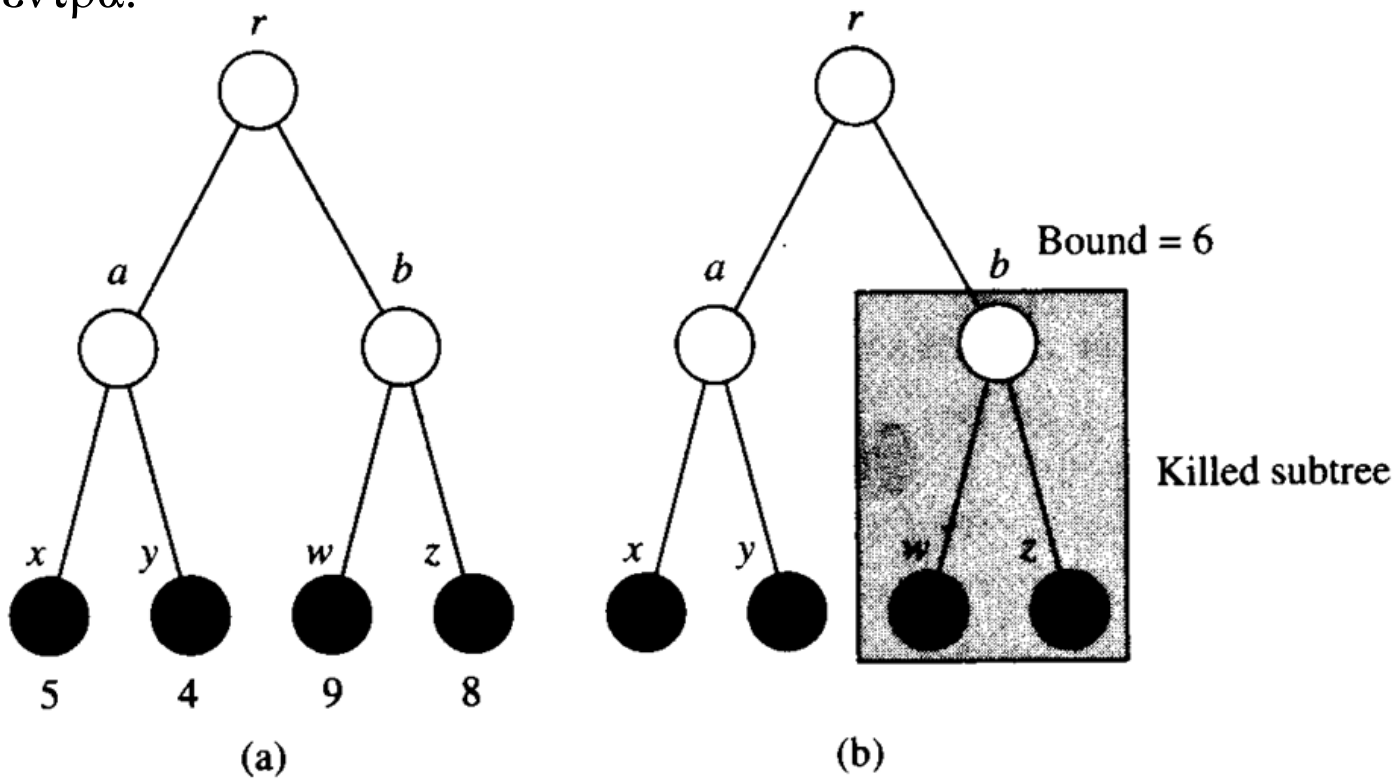
- Επίσκεψη μόνο ενός τμήματος του δέντρου.
- Για κάθε διακλάδωση (επιλογή τιμής σε μία μεταβλητή) εκτιμάται ένα κάτω όριο για όλες τις λύσεις στο υποδέντρο.
- Εάν αυτό είναι μεγαλύτερο από την καλύτερη έως τότε λύση το υποδέντρο εγκαταλείπεται γιατί κάθε λύση του είναι χειρότερη.

Worst Case: *εκθετική*, *Average Case: βιώσιμη λύση*.

Αλγόριθμος Branch & Bound

Χαρακτηριστικά: Επιλογή διακλάδωσης – Συνάρτηση υπολογισμού ορίου

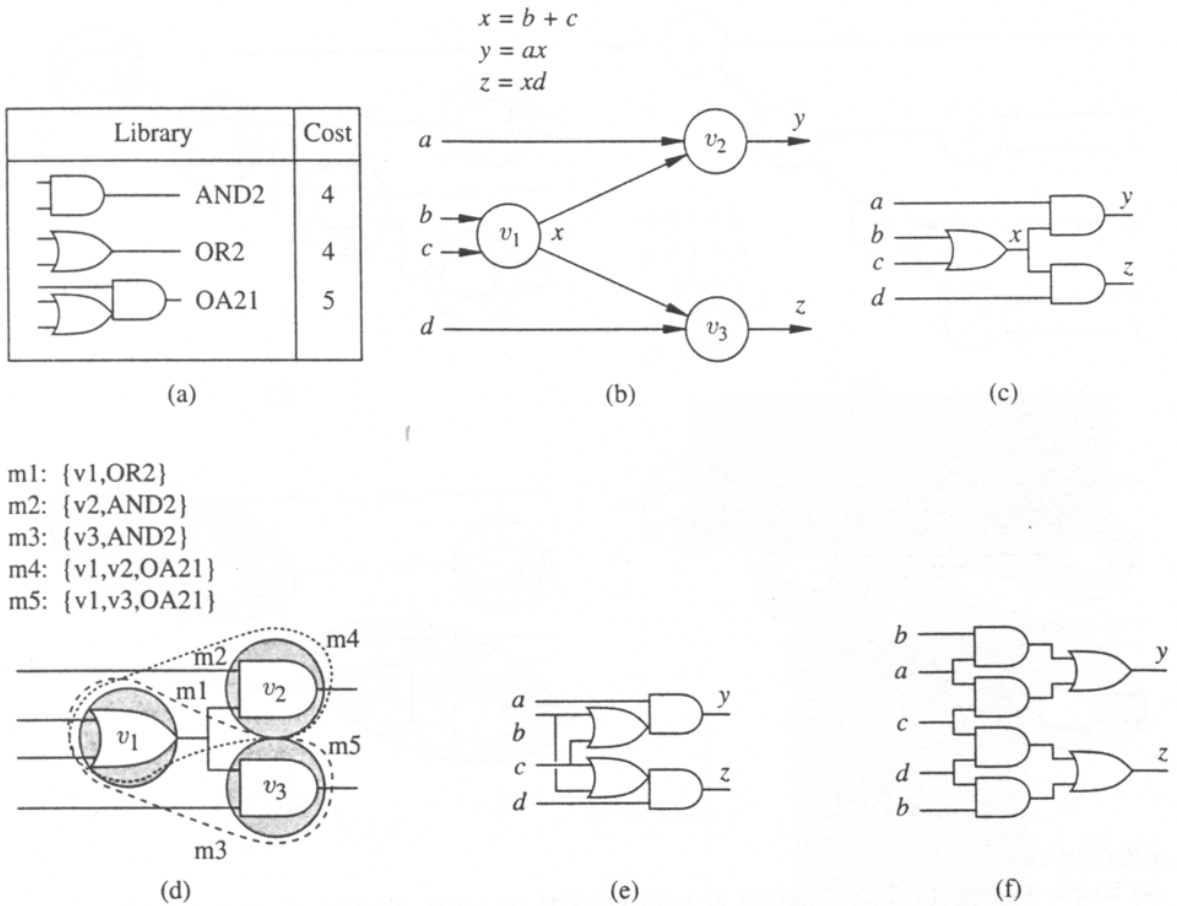
Η συνάρτηση υπολογισμού ορίου πρέπει να είναι γρήγορη και να δίνει αποτελέσματα πολύ κοντά στην βέλτιστη τιμή για να απορρίπτονται πολλά υποδέντρα.



Αλγόριθμος Branch & Bound

```
BRANCH_AND_BOUND {
    Current_best = anything;
    Current_cost = ∞;
    S = s0;
    while (S ≠ ∅) do {
        Select an element in s ∈ S;
        Remove s from S;
        Make a branch based on s yielding sequences {si, i = 1, 2, ..., m};
        for ( i = 1 to m) {
            Compute the lower bound bi of si;
            if (bi ≥ Current_cost)
                Kill si;
            else {
                if (si corresponds to a complete solution ) {
                    Current_best = si;
                    Current_cost = cost of si;
                }
                else
                    Add si to set S;
            }
        }
    }
}
```

Πρόβλημα & Ανάλυση



Διάφορα bindings με διαφορετικό κόστος

Αλγόριθμοι

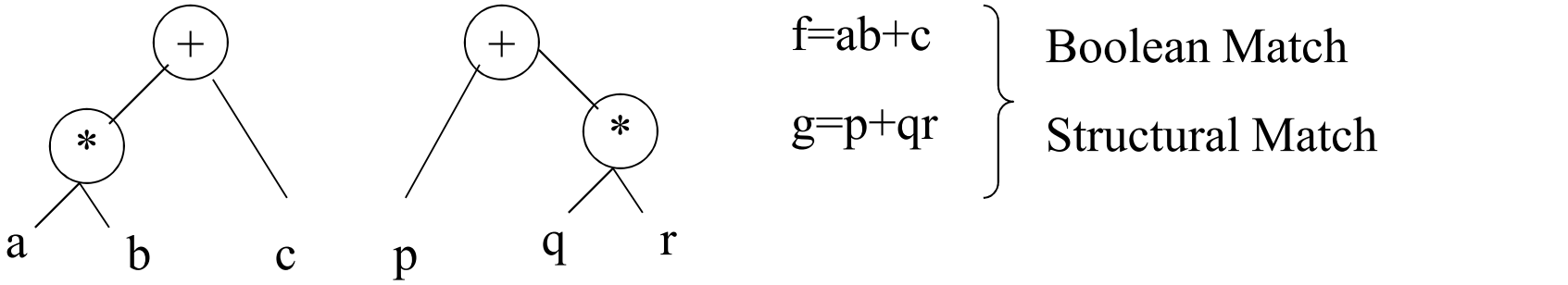
Υπάρχουν δύο προσεγγίσεις για τις αντιστοιχίσεις:

Α. Προσέγγιση Boolean: το δίκτυο και τα κύτταρα της βιβλιοθήκης αναπαρίστανται με Boolean συναρτήσεις.

Β. Προσέγγιση Κατασκευής: χρησιμοποιούνται γράφοι που αναπαριστούν αλγεβρικές αποσυνθέσεις.

Ορισμός Boolean match. Δύο συνδυαστικές συναρτήσεις μονής εξόδου είναι ισοδύναμες εάν υπάρχει μήτρα αντιμετάθεσης P για την οποία ισχύει f(x)=g(Px). Η μήτρα αντιμετάθεσης μοντελοποιεί την ελευθερία ανάθεσης pins εισόδου.

Structural match: είναι η ισομορφία γράφων.



Αλγόριθμοι

Για την απλοποίηση του προβλήματος κάλυψης οι περισσότεροι ευριστικοί αλγόριθμοι εφαρμόζουν δύο βήματα προ-επεξεργασίας: decomposition και partitioning.

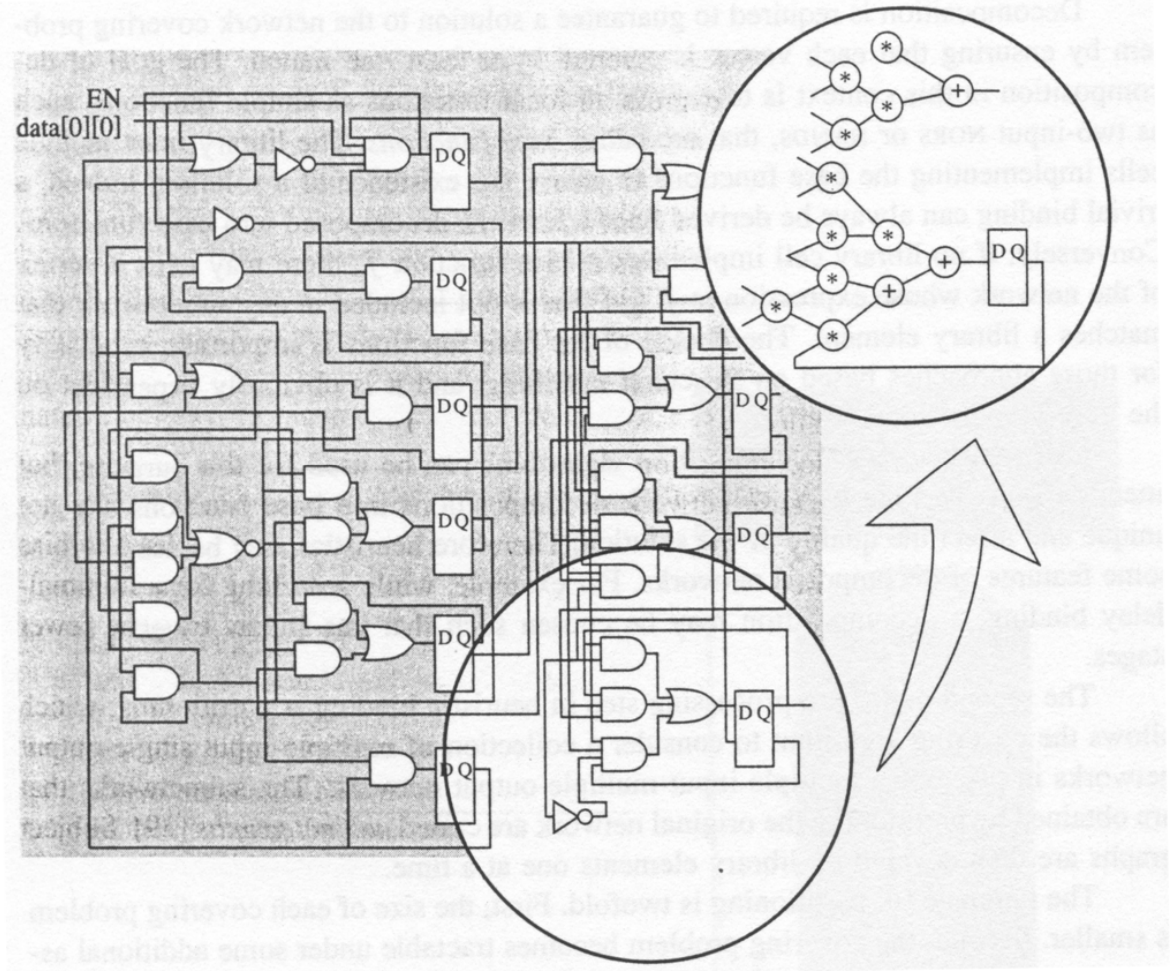
1. **Decomposition**: εγγυάται ότι κάθε κορυφή καλύπτεται από τουλάχιστον ένα ταίριασμα. Στόχος είναι η έκφραση όλων των τοπικών συναρτήσεων από βασικές συναρτήσεις (and, or, nand, nor, exor, exnor).

   Υπάρχουν πολλοί τρόποι decomposition και πρέπει να είναι κατευθυνόμενο έτσι ώστε να βελτιστοποιείται το κύκλωμα.

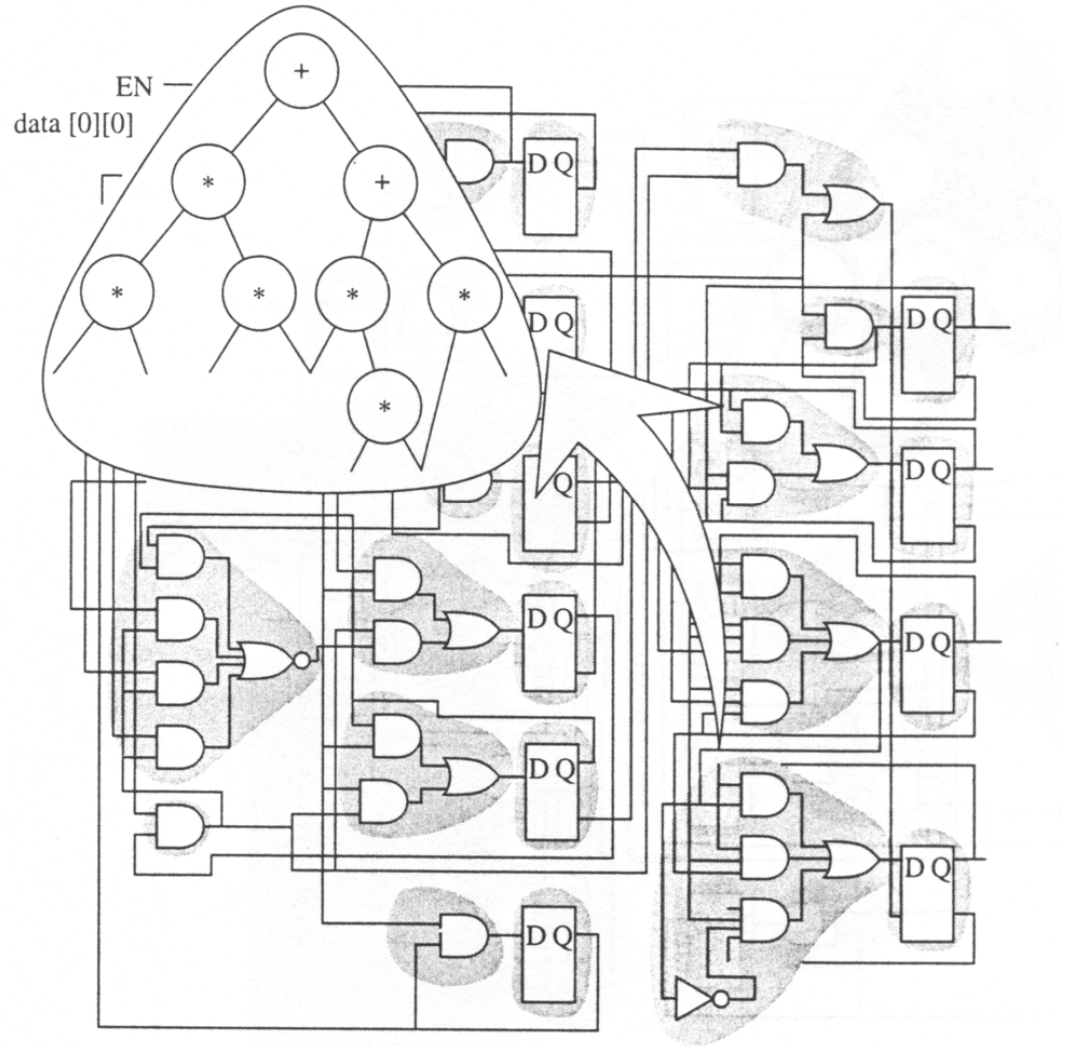
2. **Partitioning**: διαιρεί τον γράφο σε υπογράφους και κάθε φορά μελετάται ένας υπογράφος μίας εξόδου.

   Σε κάθε match το τμήμα του γράφου που ταιριάζει με κάποια κύτταρα της βιβλιοθήκης παίρνει μία ετικέτα μαζί με ιδιότητες επιφάνειας και καθυστέρησης.

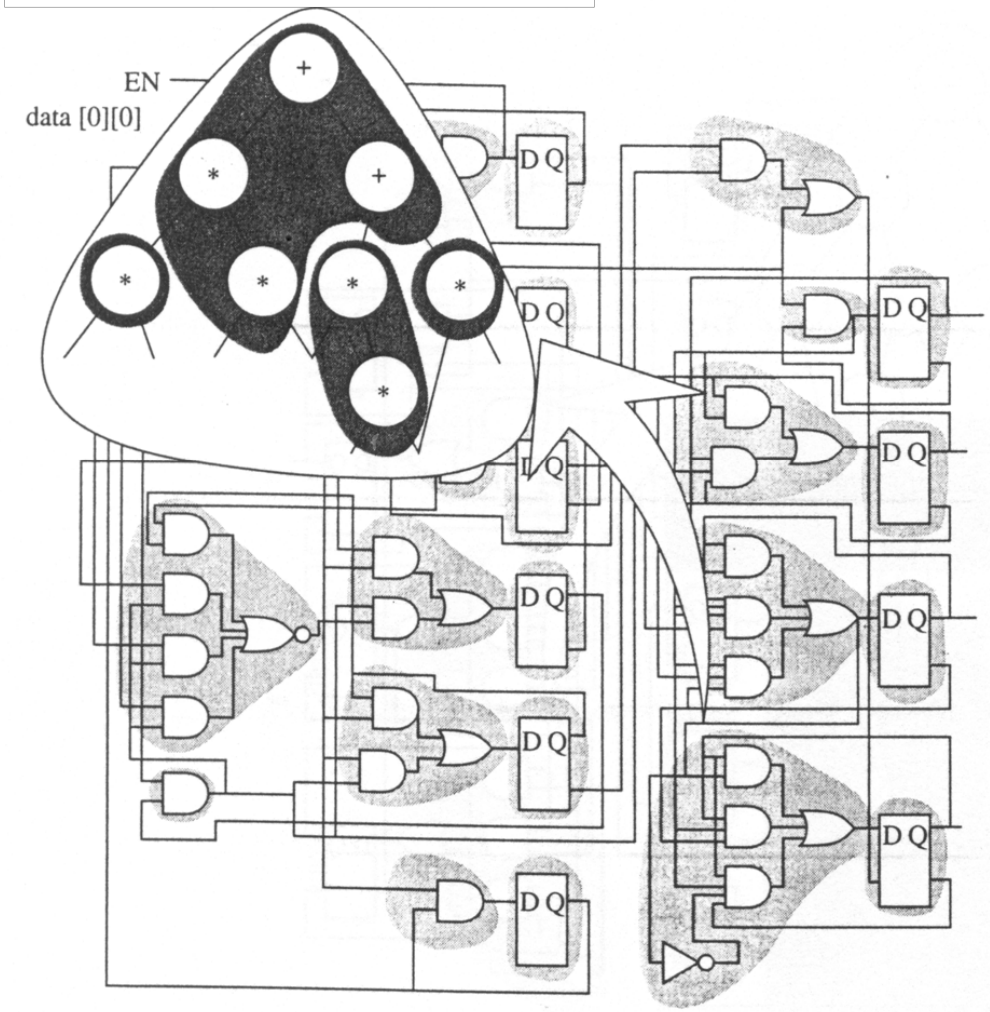
Decomposition



Partitioning



Covering



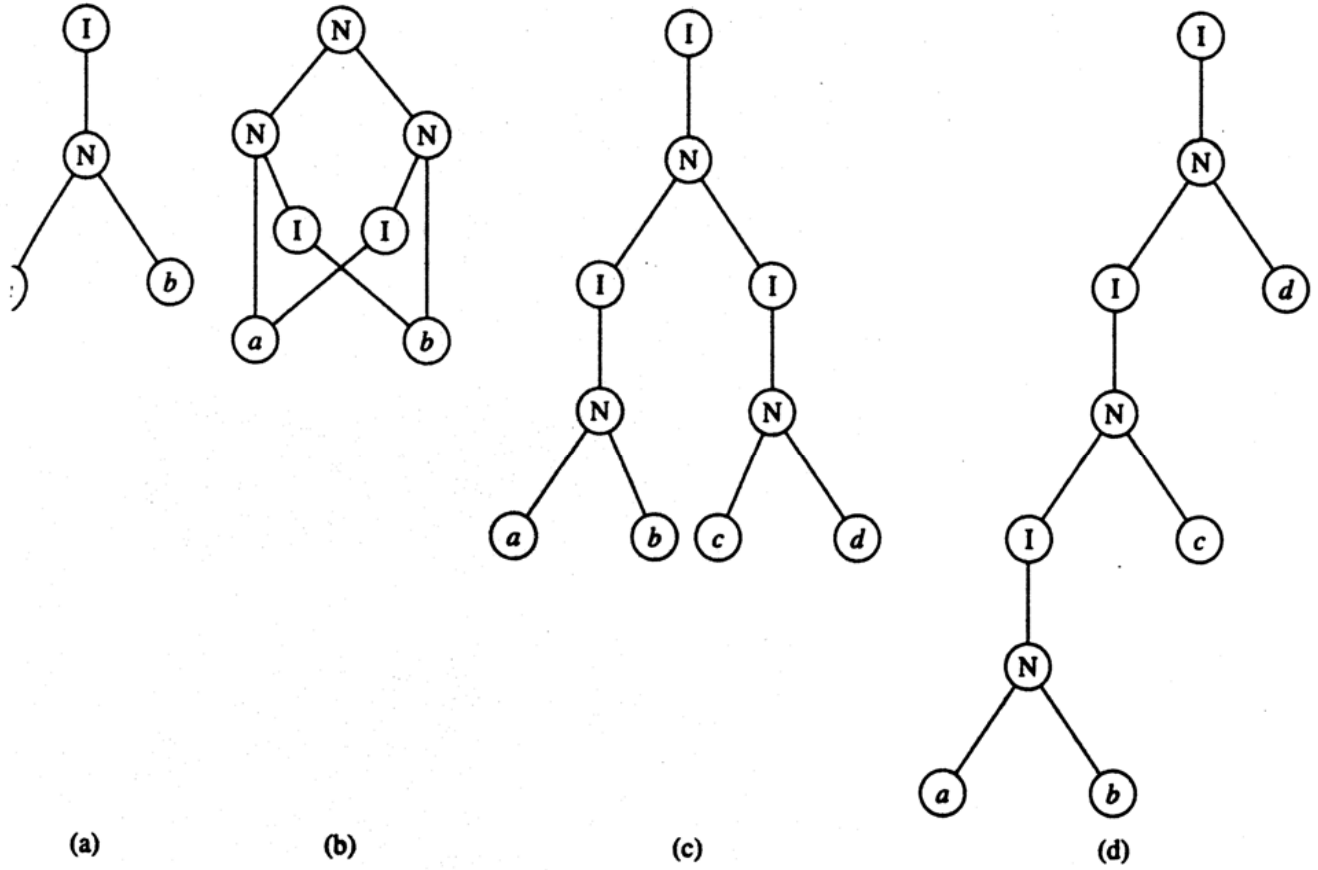
Covering με structural matching

- ✓ Βασίζεται στην αναγνώριση κοινών patterns. Γράφος και συναρτήσεις βιβλιοθήκης αποσυντίθενται σε βασικές συναρτήσεις.
- ✓ **Pattern graphs**: οι γράφοι που σχετίζονται με τα στοιχεία της βιβλιοθήκης
- ✓ **Subject graphs**: οι γράφοι που αντιστοιχίζονται στην βιβλιοθήκη.
- ✓ Οι γράφοι subject/pattern: είναι άκυκλοι και έχουν ρίζα.
- ✓ Θεωρούμε ότι το decomposition οδηγεί σε

  (α) δέντρα και

  (β) **leaf-dags**: άκυκλοι γράφοι όπου τα μονοπάτια από την ρίζα (έξοδος) συγκλίνουν μόνο σε φύλλα (είσοδοι).
- ✓ Οι αντιστροφείς μοντελοποιούνται ρητά.
- ✓ Το structural matching ελέγχει την ισομορφικότητα μεταξύ δύο dags.

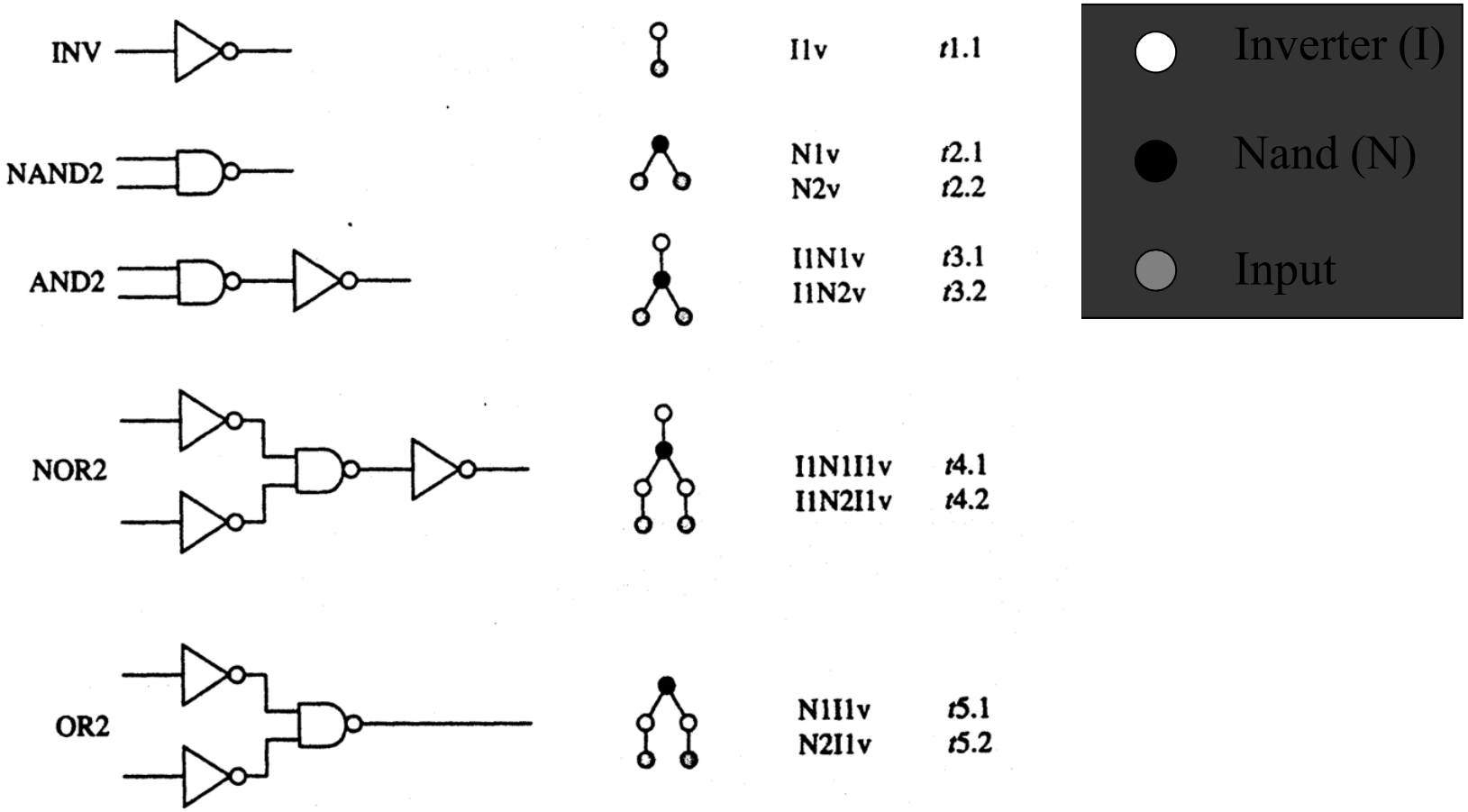
Covering με structural matching



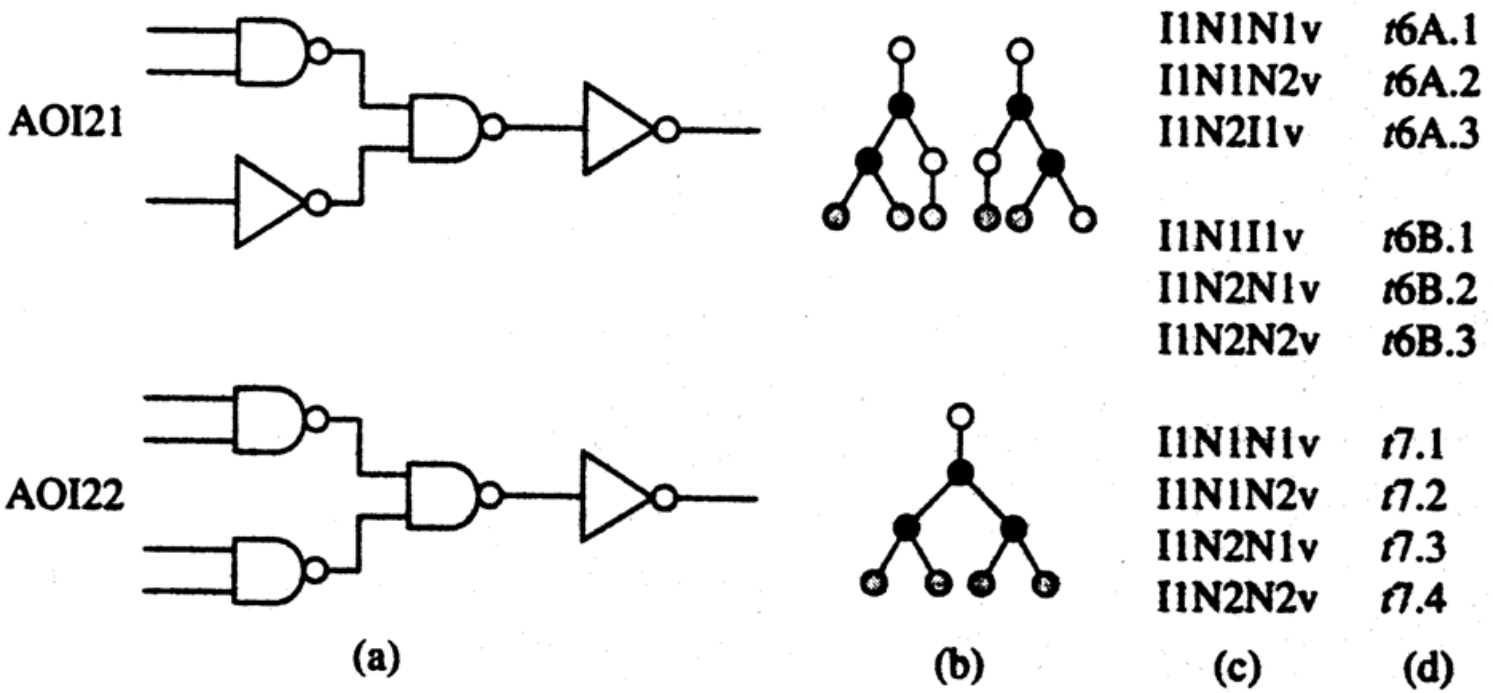
a, c, d: trees – b: leaf dag

Covering με structural matching

Υπόθεση: χρήση πυλών NAND – NOT για υλοποίηση κάθε λογικής



Covering με structural matching



Ταίριασμα βασισμένο σε δέντρα (απλό)

- ✓ Θεωρούμε ότι μόνο μια βασική συνάρτηση χρησιμοποιείται στην decomposition (2-input nand).
- ✓ Κάθε κορυφή του δέντρου σχετίζεται με μία nand 2 εισόδων και έχει 2 παιδιά, ή με αντιστροφέα και έχει 1 παιδί.
- ✓ Ελέγχουμε εάν ένα pattern tree είναι ισομορφικό με έναν υπογράφο του subject tree. Αυτό επιτυγχάνεται με

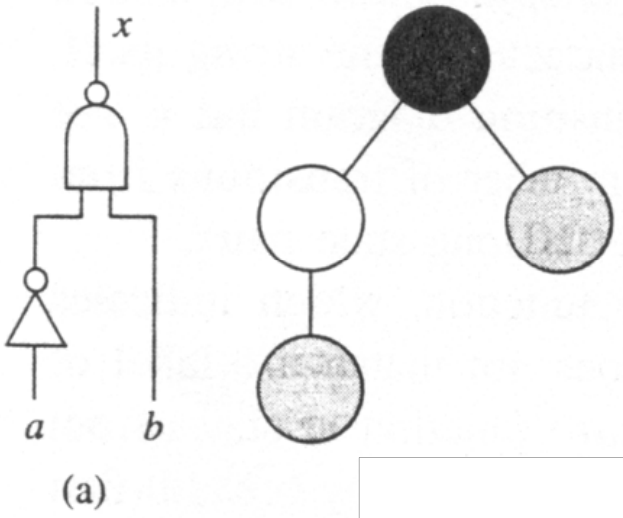
  A. Ταίριασμα της ρίζας του pattern tree με μία κορυφή του subject tree και
  B. Αναδρομική επίσκεψη των παιδιών τους.
- ✓ Ο έλεγχος ταιριάσματος είναι ουσιαστικά η ισότητα του αριθμού παιδιών σε κάθε κορυφή που ελέγχεται (ο τύπος όλων των κορυφών είναι ίδιος).
- ✓ Όταν φτάσουμε σε φύλλο στο pattern tree τότε έχουμε ταίριασμα.
- ✓ Όταν φτάσουμε σε φύλλο στο subject tree και σε μη-φύλλο στο pattern tree τότε έχουμε αδυναμία ταιριάσματος.

Ταίριασμα βασισμένο σε δέντρα (απλό)

Pattern : u, Subject : v

```
MATCH(u, v) {
    if (u is a leaf) return (TRUE);                          /* Leaf of the pattern graph reached */
    else {
        if (v is a leaf) return (FALSE);                     /* Leaf of the subject graph reached */
        if (degree(v) ≠ degree(u)) return(FALSE);                          /* Degree mismatch */
        if (degree(v) == 1) {                          /* One child each: visit subtree recursively */
            u_c = child of u ; v_c = child of v ;
            return (match(u_c, v_c) )
        }
        else {                                         /* Two children each: visit subtrees recursively */
            u_l = left-child of u ; u_r = right-child of u ;
            v_l = left-child of v ; v_r = right-child of v ;
            return (MATCH(u_l, v_l) · MATCH(u_r, v_r) + MATCH(u_r, v_l) · MATCH(u_l, v_r));
        }
    }
}
```

Ταίριασμα βασισμένο σε δέντρα (απλό)



(a)

Δυναμικός Προγραμματισμός

- ✓ Αλγοριθμική μέθοδος που λύνει ένα πρόβλημα βελτιστοποίησης με την **διαίρεσή** του σε ακολουθία αποφάσεων.
- ✓ Δίνει βέλτιστη λύση όταν το ίδιο το πρόβλημα έχει **βέλτιστη θεμελίωση** δηλ. η βέλτιστη λύση του προβλήματος περιέχει βέλτιστες λύσεις για τα υποπροβλήματα του.

Η αποδοτικότητα του αλγόριθμου εξαρτάται από το **μήκος** της ακολουθίας απόφασης και την **πολυπλοκότητα λύσης** των υποπροβλημάτων.

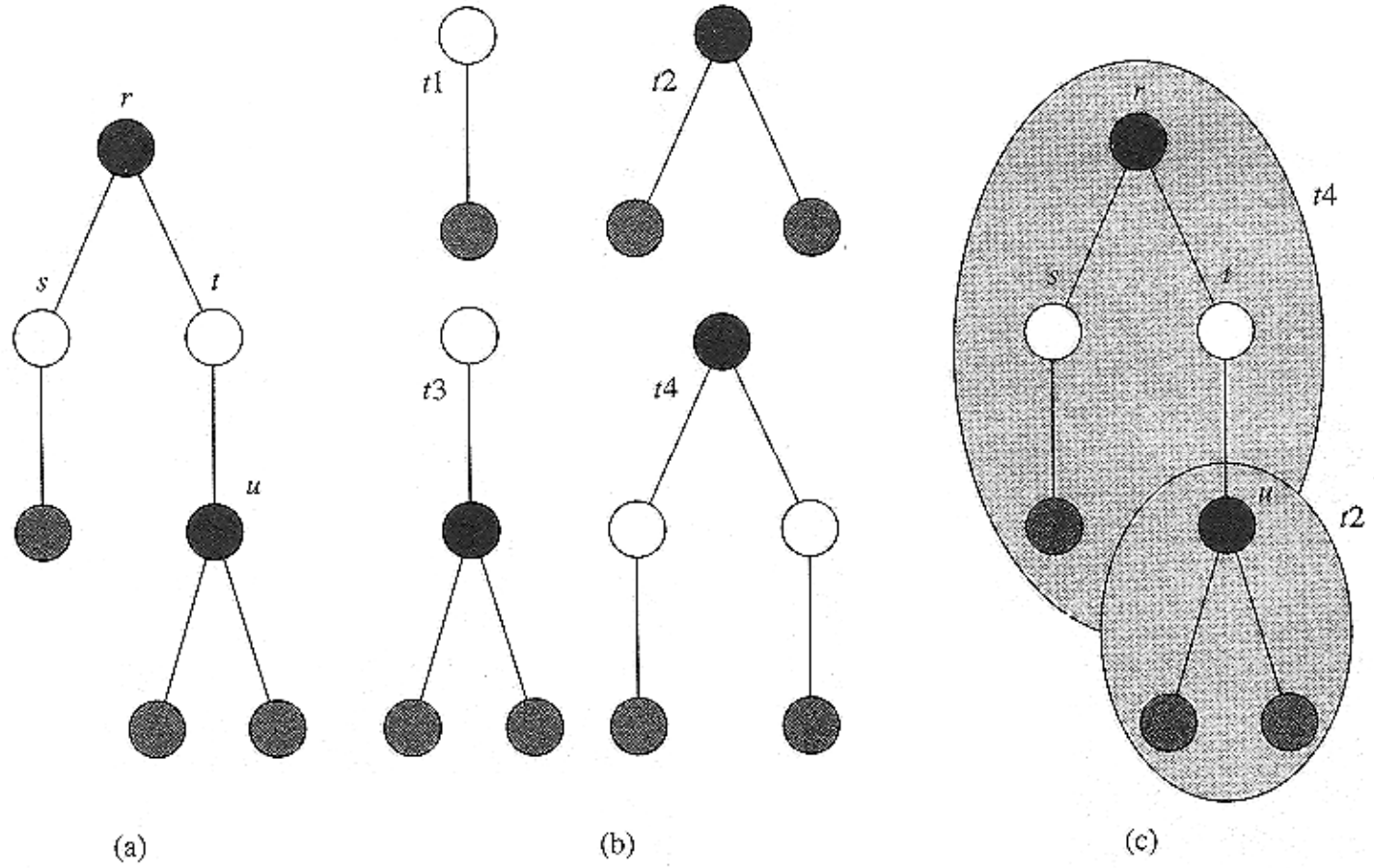
Παράδειγμα: κάλυψη δέντρου (subject) από πρότυπα υποδέντρα (patterns).

- ➢ Κάθε κορυφή είναι και μία απόφαση και τις περνάμε bottom-up.
- ➢ Κάθε κορυφή έχει ένα κόστος με τα φύλλα να έχουν κόστος 0.

Δυναμικός Προγραμματισμός

- Για κάθε κορυφή ελέγχουμε εάν το υποδέντρο είναι ισομορφικό με κάποιο από τα πρότυπα δέντρα.
- Τότε το κόστος της κορυφής είναι ίσο με το κόστος του ισομορφικού δέντρου και το άθροισμα των αντίστοιχων φύλλων του (κορυφές υποδέντρων ή τελικά φύλλα).
- Επιλογή του ισομορφικού δέντρου με το μικρότερο κόστος.

Δυναμικός Προγραμματισμός



Κόστος πρότυπων δέντρων: t_1=2, t_2=3, t_3=4, t_4=5

Δυναμικός Προγραμματισμός

```
TREE_COVER(T(V, E)) {
    Set the cost of the internal vertices to −1;
    Set the cost of the leaf vertices to 0;
    while (some vertex has negative weight) do {
        Select a vertex v ∈ V whose children have all nonnegative cost;
        M = set of all matching pattern trees at vertex v;
        cost (v) = min_{m∈M(v)} (cost(m) + Σ_{u∈L(m)} cost(u) );
    }
}
```

✓Πολυπλοκότητα O(|V|): αριθμός αποφάσεων όσες οι κορυφές

✓Για κάθε κορυφή υπάρχει ένα όριο συγκρίσεων που εξαρτάται από τον αριθμό των pattern trees.

✓L(m): Για κάθε υποδέντρο που εξετάζεται είναι το σύνολο κορυφών που αντιστοιχούν στα φύλλα του ισομορφικού πρότυπου δέντρου.

Βέλτιστη Κάλυψη βασισμένη σε δέντρα

- Η βέλτιστη κάλυψη δέντρου υπολογίζεται με δυναμικό προγραμματισμό.
- Κάθε κύτταρο έχει ένα κόστος επιφάνειας.
- Η συνολική επιφάνεια του δικτύου είναι το αντικείμενο της ελαχιστοποίησης.
- Ο αλγόριθμος κάλυψης του δέντρου το διαπερνάει από κάτω προς τα πάνω.
- Ελέγχει το ταίριασμα κάθε pattern tree θεωρώντας την αντιστοιχία μιας κορυφής του δικτύου με την ρίζα του pattern tree.

Βέλτιστη Κάλυψη βασισμένη σε δέντρα

- Για κάθε κορυφή του subject tree ο αλγόριθμος ελέγχει το ταίριασμα του υποδέντρου με ρίζα την κορυφή, με τα pattern trees. Οι πιθανότητες είναι:

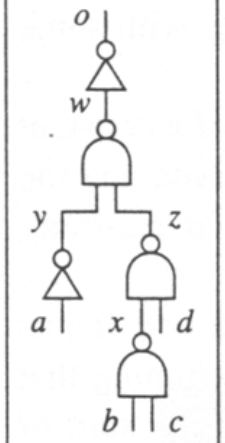
  1. Το pattern tree και το subject subtree είναι ισομορφικά. Τότε το κόστος του κυττάρου είναι και κόστος της κορυφής.

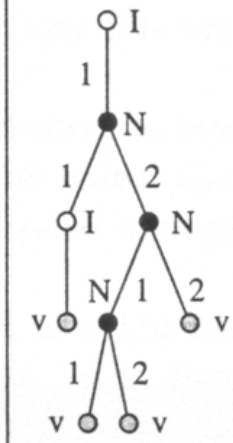
  2. Το pattern tree είναι ισομορφικό τμήμα του subject subtree. Τότε η κορυφή παίρνει σαν κόστος το κόστος του κυττάρου και το κόστος καθενός από τα φύλλα του τμήματος του subject subtree.

  3. Δεν υπάρχει ταίριασμα.

Βέλτιστη Κάλυψη βασισμένη σε δέντρα

- Εάν η βιβλιοθήκη καλύπτει τις βασικές συναρτήσεις τότε για κάθε κορυφή υπάρχει τουλάχιστον ένα κύτταρο που ικανοποιεί την 1 ή 2.
- Για κάθε κορυφή επιλέγουμε το καλύτερο ταίριασμα.
- Στο τέλος κάθε περάσματος οι ετικέτες αντιστοιχούν στην βέλτιστη κάλυψη.





Vertex	Match	Gate	Cost
x	*t*2	NAND2(*b*,*c*)	NAND2
y	*t*1	INV(*a*)	INV
z	*t*2	NAND2(*x*,*d*)	2 NAND2
w	*t*2	NAND2(*y*,*z*)	3 NAND2+INV
o	*t*1	INV(*w*)	3 NAND2+2INV
	*t*3	AND2(*y*,*z*)	2 NAND2+AND2+INV
	*t*6*B*	AOI21(*x*,*d*,*a*)	NAND2+AOI21

Ελαχιστοποίηση καθυστέρησης

- Στόχος είναι η ελαχιστοποίηση των data ready χρόνων.
- Το κόστος κάθε κυττάρου είναι η καθυστέρηση διάδοσης εισόδου/εξόδου.
- Η καθυστέρηση μετάδοσης ενός κυττάρου μπορεί να θεωρηθεί σταθερή η εξαρτώμενη από τα Fanin, Fanout.
- Η συνολική καθυστέρηση ενός δικτύου ισούνται με την καθυστέρηση του κρίσιμου μονοπατιού.
- Ο data-ready χρόνος στην έξοδο κάθε κυττάρου είναι ίσος με τον μεγαλύτερο από τους data-ready χρόνους των εισόδων του συν την καθυστέρηση διάδοσής του.
- Η διαπέραση από κάτω προς τα πάνω επιτρέπει την εύρεση της δέσμευσης που ελαχιστοποιεί τον data-ready χρόνο σε κάθε κορυφή, και άρα τον ελάχιστο χρόνο στην ρίζα. Ο αλγόριθμος είναι ίδιος, ενώ αλλάζει το κόστος κάθε κορυφής.

Αλγόριθμος Κάλυψης

Μειονεκτήματα αλγόριθμου κάλυψης

1. Κάθε κορυφή του subject graph θα πρέπει να ελεγχθεί για ταίριασμα έναντι ενός μεγάλου αριθμού pattern graphs.
2. Κάποια κύτταρα (ExOrs, ExNors) δεν αναπαρίστανται με δέντρα.
3. Το structural matching μπορεί να ανιχνεύσει μικρό μόνο αριθμό πιθανών ταιριασμάτων γιατί δεν λαμβάνει υπόψη αδιάφορους όρους.

Binding σε FPGAs

- Τα FPGAs είναι προ-καλωδιωμένα κυκλώματα που προγραμματίζονται από τους χρήστες.
- Διαιρούνται σε δύο κατηγορίες προγραμματισμού:

  (α) soft (Look Up Tables) και

  (β) hard (antifuses).
- Το binding σε προκαλωδιωμένα κυκλώματα είναι αρκετά δύσκολο αφού εξαρτάται από τον φυσικό σχεδιασμό τους.
- Οι καθυστερήσεις των μονοπατιών εξαρτώνται κατά πολύ από τις καλωδιώσεις εξαιτίας της τεχνολογίας προγραμματιζόμενων διασυνδέσεων.
- Κάθε LookUp Table n εισόδων μπορεί να υλοποιήσει ${}_2 2^n$ συναρτήσεις οι οποίες δεν μπορούν να απαριθμηθούν ρητά από καμία βιβλιοθήκη.

Binding σε FPGAs

Πρόβλημα:

δοσμένου ενός λογικού δικτύου ζητάμε την

εύρεση ενός ισοδύναμου λογικού δικτύου

με ελάχιστο αριθμό κορυφών (ή ελάχιστη καθυστέρηση μονοπατιών)

έτσι ώστε κάθε κορυφή να αντιστοιχεί σε μία συνάρτηση

υλοποιήσιμη από ένα LookUp Table.

Αντιμετώπιση

- Αρχικά εφαρμόζεται decomposition του λογικού δικτύου σε βασικές πύλες.
- Κατόπιν γίνεται προσπάθεια κάλυψης όσο το δυνατόν περισσότερης λογικής σε κάθε Lookup Table:

(α) θεωρούμε ένα άθροισμα παραγόντων για μία συνάρτηση μίας εξόδου, με κάθε παράγοντα να έχει n μεταβλητές.

Binding σε FPGAs

(β) ομάδες παραγόντων θα πρέπει να ανατεθούν σε διαφορετικούς πίνακες.

Παράδειγμα

Έστω η συνάρτηση f=ab+cd που πρέπει να υλοποιηθεί με LUTs για n=3. Τότε με το decomposition: $f=f_1+f_2$, $f_1=ab$, $f_2=cd$ απαιτούνται 3 LUTs, ενώ με το decomposition: $f=ab+f_2$, $f_2=cd$ απαιτούνται 2 LUTs

- Λύση:

  1. Επιλέγει τον παράγοντα με τις περισσότερες μεταβλητές και τον τοποθετεί σε έναν LUT. Εάν δεν αρκεί ένα LUT προστίθενται και άλλα.

  2. Όταν όλοι οι παράγοντες έχουν ανατεθεί σε LUTs τότε ο πίνακας με τις λιγότερες αχρησιμοποίητες μεταβλητές ορίζεται τελικός, παίρνει μία μεταβλητή και ανατίθεται στον πρώτο πίνακα που μπορεί να τον δεχτεί.

Binding σε FPGAs

3. Όταν μείνει ένας μόνο πίνακας τερματίζει ο αλγόριθμος.

FPGAs βασισμένα σε AntiFuses

- Η νοητή βιβλιοθήκη αποτελείται από όλες τις συναρτήσεις που μπορεί να υλοποιήσει ένα logic module πχ. Act1 series:

$$m_1=(s_0+s_1)(s_2a+s_2'b)+s_0's_1'(s_3c+s_3'd)$$

- Binding: δεδομένου ενός συνδυαστικού λογικού δικτύου να βρεθεί ένα ισοδύναμο με ελάχιστο αριθμό κορυφών (ή καθυστέρησης μονοπατιών) τέτοιων ώστε κάθε μία να μπορεί να υλοποιηθεί από το logic module του FPGA.

Binding σε FPGAs

- Όταν οι συναρτήσεις που μπορεί να υλοποιήσει κάθε logic module δεν είναι υπερβολικά πολλές, είναι καλύτερο να τις απαριθμούμε γιατί έτσι μπορούν να χρησιμοποιηθούν οι κλασσικοί αλγόριθμοι binding.
- Όταν αυτό δεν είναι εφικτό θα πρέπει να χρησιμοποιηθούν structural και boolean τεχνικές.
- Στις structural τεχνικές πρέπει να διερευνάται η υλοποίηση του module και να εκμεταλλευόμαστε τις ιδιότητες του. Πχ όταν υλοποιείται με πολυπλεξία μπορούμε να κάνουμε το decomposition με βασικά στοιχεία πολυπλέκτες.

Binding με κανόνες

- Το binding γίνεται βηματικά με τοπικούς μετασχηματισμούς που διατηρούν την συμπεριφορά του.
- Κάθε μετασχηματισμός είναι η αντικατάσταση ενός υποδικτύου με ένα ισοδύναμο από την βιβλιοθήκη.
- Κάθε στοιχείο της data base των κανόνων περιέχει ένα λογικό pattern και ένα ισοδύναμό του από την βιβλιοθήκη.
- Κάθε στοιχείο μπορεί να κωδικοποιεί απλούς ή περίπλοκους κανόνες. Οι απλοί ορίζουν ένα καλό ταίριασμα για το υποδίκτυο. Οι περίπλοκοι ορίζουν μια αναδόμηση του δικτύου.
- Μπορεί να επιλεγεί ο κανόνας που βελτιστοποιεί τοπικά το δίκτυο με βάση κάποια μετρική κόστους.

Binding με κανόνες