ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και Δικτύων

Διπλωματική Εργασία

Θέμα: «Τεχνικές χωροθέτησης ολοκληρωμένων κυκλωμάτων»

Όνομα: Νικόλαος Μιχαηλίδης

Α.Ε.Μ.: 44

Επιβλέποντες:

Σταμούλης Γεώργιος, Καθηγητής

Μποζάνης Παναγιώτης, Αναπληρωτής Καθηγητής

Βόλος, Οκτώβριος 2012

Ευχαριστίες

Θα ήθελα, αρχικά, να ευχαριστήσω θερμά τον Δρ. Σταμούλη Γεώργιο, καθηγητή του τμήματος Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων, κύριο επιβλέποντα της διπλωματικής μου εργασίας για την συμπαράσταση και την εμπιστοσύνη που μου έδειξε κατά την εκπόνηση της διπλωματικής εργασίας, αλλά και καθ' όλη την διάρκεια των σπουδών μου.

Επίσης, θα ήθελα να εκφράσω τις ευχαριστίες μου και την ειλικρινή μου εκτίμηση στο πρόσωπο του προέδρου του τμήματος Δρ. Μποζάνη Παναγιώτη για την έμπρακτη υποστήριξη του και την καίρια συμβολή του στην ολοκλήρωση των σπουδών μου.

Από καρδίας θέλω να εκφράσω την ευγνωμοσύνη μου στον Δρ. Δαδαλιάρη Αντώνιο για την καθοδήγησή του στην εκπόνηση της εργασίας, την απέραντη υπομονή του, την αμέριστη υποστήριξη που μου χάρισε και την πίστη του – μεγαλύτερη και από την δική μου - στην απόκτηση του διπλώματός μου.

Τέλος, θέλω να ευχαριστήσω την σύζυγό μου, την οικογένειά μου και όλους όσοι με οποιονδήποτε τρόπο συνέβαλαν στην επίτευξη του στόχου μου.

Περιεχόμενα

Το πρόβλημα της τοποθέτησης κελιών VLSI είναι γνωστό ότι είναι ένα πρόβλημα NP (σε μη-ντετερμινιστικό πολυωνυμικό χρόνο). Ένα ευρύ φάσμα ευρετικών αλγορίθμων υπάρχει στην βιβλιογραφία για την αποτελεσματική διευθέτηση των λογικών πυλών σε ένα τσιπ VLSI. Ο στόχος της παρούσας εργασίας είναι να παρουσιάσει μια ολοκληρωμένη μελέτη των διαφόρων τεχνικών τοποθέτησης κελιών, με έμφαση στις standard cell και macro placement. Πέντε βασικοί αλγόριθμοι για τοποθέτηση κελιών θα συζητηθούν: simulated annealing, force-directed placement, min-cut placement και evolution-based placement. Οι δύο πρώτες κλάσεις αλγορίθμων οφείλουν την ονομασία τους στους φυσικούς νόμους, η τρίτη κλάση είναι αναλυτικές τεχνικές, και η τέταρτη κλάση των αλγορίθμων προέρχεται από βιολογικά φαινόμενα. Σε κάθε κατηγορία, ο βασικός αλγόριθμος εξηγείται με κατάλληλα παραδείγματα. Επίσης θα συζητήσουμε τις εφαρμογές των αλγορίθμων.

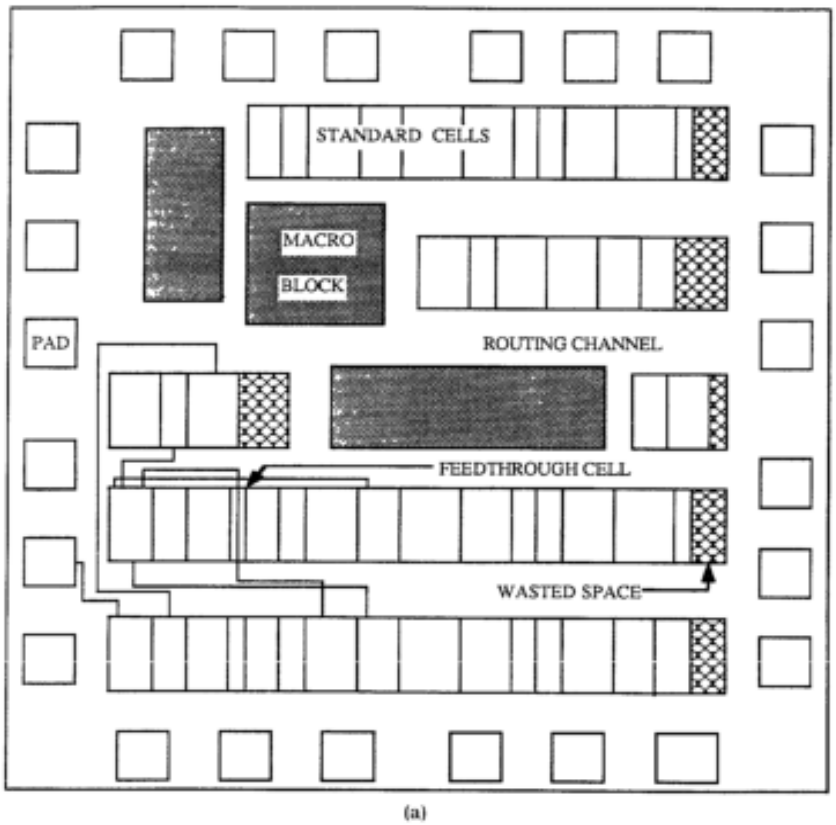
Γενικοί Όροι: Σχεδίαση, Επιδόσεις

Πρόσθετες λέξεις και φράσεις κλειδιά: VLSI, τοποθέτηση (placement), διάταξη (layout), φυσικός σχεδιασμός (physical design), χωροθέτηση (floor planning), simulated annealing, ολοκληρωμένα κυκλώματα (integrated circuit), γενετικοί αλγόριθμοι (genetic algorithms), αναγκαστικά οδηγούμενη τοποθέτηση (force-directed placement), min-cut, πίνακας πυλών (gate array), πρότυπο κελί (standard cell).

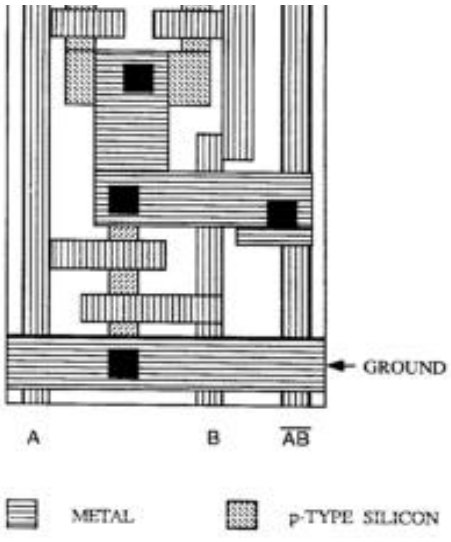
Εισαγωγή

Τα εργαλεία σχεδίασης στοχευμένα για υπολογιστές είναι ικανά πλέον να αυτοματοποιήσουν όλη τη διαδικασία εξόδου που ακολουθεί το στάδιο του σχεδιασμού του ολοκληρωμένου κυκλώματος. Αυτό κατέστη δυνατό κυρίως με τη χρήση πίνακα πυλών (gate array) και με το σχεδιασμό standard cell, σε συνδυασμό με έξυπνα πακέτα λογισμικού για αυτόματη τοποθέτηση και δρομολόγηση. Το Σχήμα 1a δείχνει ένα τσιπ που χρησιμοποιεί την διάταξη standard cell, το οποίο περιλαμβάνει κάποια μακρό μπλοκ (macro blocks). Τα πρότυπα στοιχεία (standard cells - Εικόνα 1b) είναι λογικές μονάδες με μια προσχεδιασμένη εσωτερική διάταξη. Έχουν σταθερό ύψος, αλλά διαφορετικά πλάτη, ανάλογα με τη λειτουργικότητα των ενοτήτων (modules). Τοποθετούνται σε γραμμές, με κανάλια δρομολόγησης ή κενά ανάμεσα στις σειρές και προορίζονται για τη διευθέτηση της διασύνδεσης μεταξύ των συνιστωσών του τσιπ. Τα τυπικά κελιά συνήθως σχεδιάζονται έτσι ώστε η διασύνδεση της ισχύς και της γείωσης να είναι οριζόντια στο άνω και κάτω άκρο του κελιού. Όταν τα κελιά τοποθετούνται δίπλα το ένα στο άλλο, οι διασυνδέσεις δημιουργούν μια συνεχή διαδρομή σε κάθε γραμμή. Οι λογικές είσοδοι και έξοδοι του κελιού είναι διαθέσιμες στις ακίδες (pins) ή στα τερματικά κατά μήκος του άνω ή του κάτω άκρου (ή και τα δύο). Είναι συνδεδεμένες μέσω έτοιμων διασυνδέσεων ή καλωδίων μέσα από τους διαύλους δρομολόγησης. Οι συνδέσεις από μία γραμμή στην άλλη γίνονται είτε μέσω των καθέτων διαύλων καλωδίωσης στις ακμές του τσιπ είτε χρησιμοποιώντας κελιά feed-through, που είναι κελιά ορισμένου ύψους με διασυνδέσεις που τα διατρέχουν κάθετα. Τα Macro blocks είναι λογικές μονάδες όχι στην τυπική μορφή των κελιών, συνήθως μεγαλύτερα από τα standard cell, και τοποθετούνται σε οποιαδήποτε βολική τοποθεσία του ολοκληρωμένου.

Εικόνα (1a) Διάταξη standard cell με macro blocks



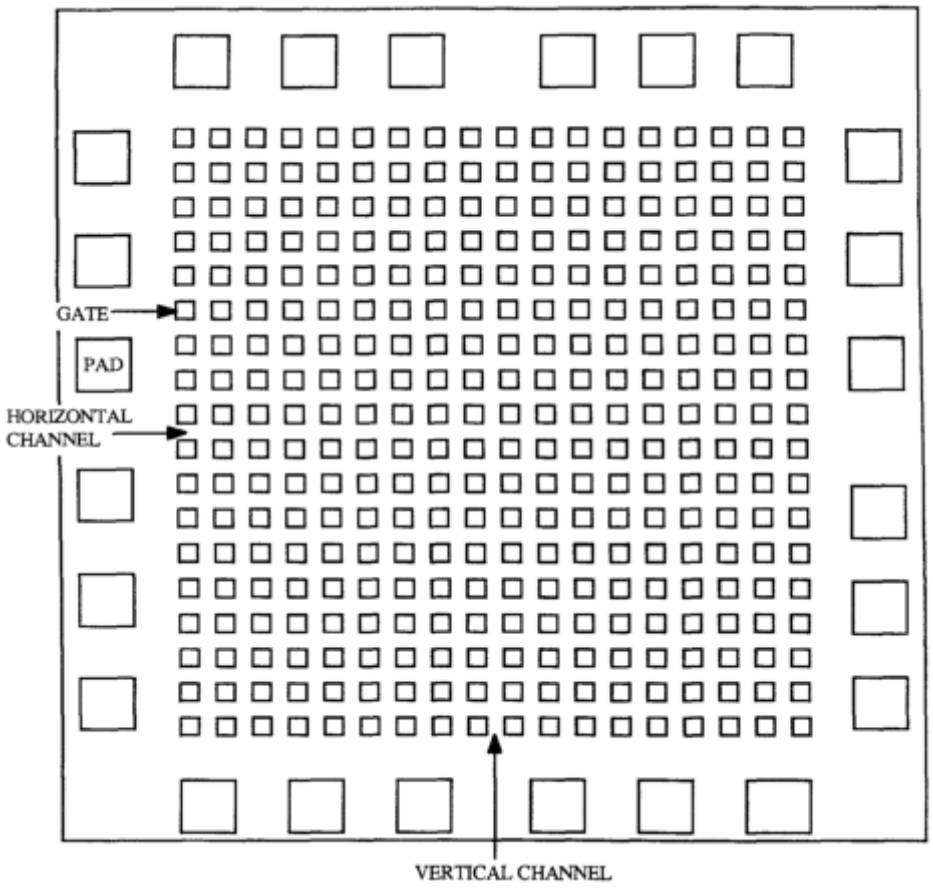
Εικόνα (1b) Διάταξη standard cell με macro blocks



Η Εικόνα 2 δείχνει ένα τσιπ που χρησιμοποιεί σχεδιασμό με gate array. Εδώ το κύκλωμα αποτελείται μόνο από θεμελιώδεις λογικές πύλες, όπως η πύλη NAND, όχι μόνο προσχεδιασμένες αλλά προκατασκευασμένες ως μία ορθογώνια συστοιχία, με

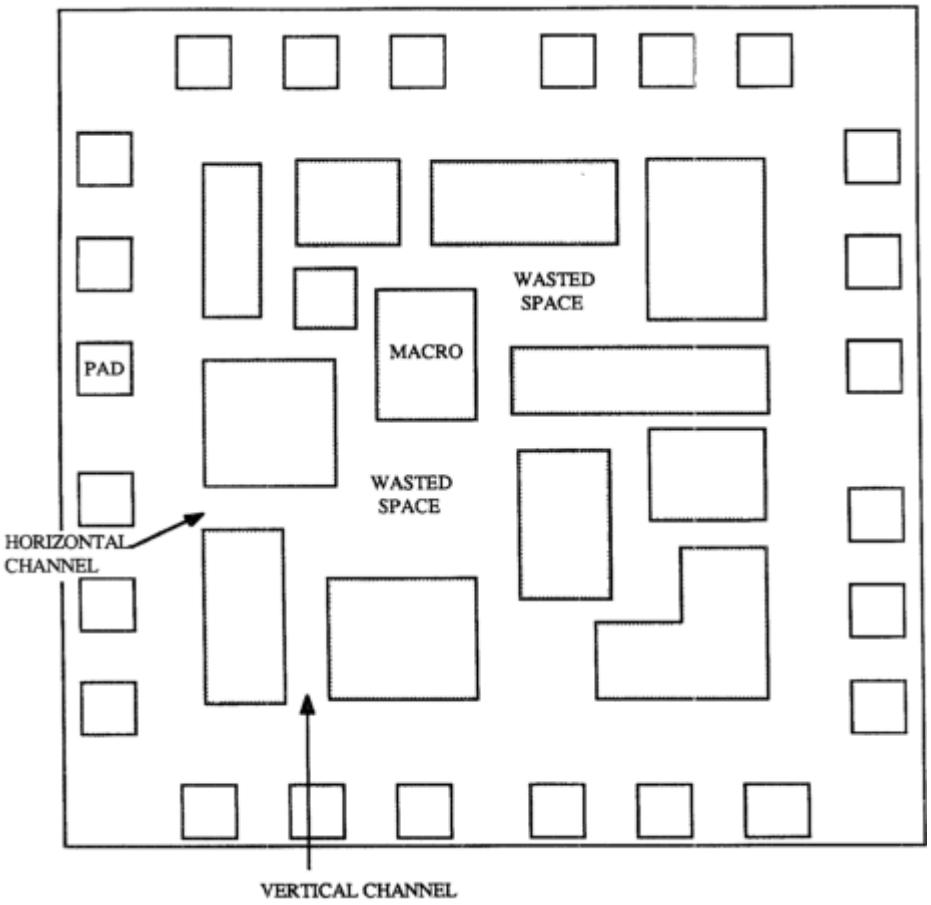
οριζόντια και κάθετα κανάλια δρομολόγησης μεταξύ των πυλών αποκλειστικά για τις διασυνδέσεις. Έτσι, ο σχεδιασμός ενός τσιπ ανάγεται στο σχεδιασμό των διασυνδέσεων σύμφωνα με το διάγραμμα του κυκλώματος. Ομοίως, η κατασκευή ενός προσαρμοσμένου (customized) τσιπ απαιτεί μόνο τα στάδια masking για τη διάταξη της διασύνδεσης.

Εικόνα 2 Διάταξη gate array



Το Σχήμα 3 δείχνει ένα τρίτο στυλ διάταξης τσιπ, το οποίο χρησιμοποιεί μόνο macro blocks. Αυτά τα μπλοκ μπορεί να έχουν ακανόνιστα σχήματα και μεγέθη, να μην ταιριάζουν μεταξύ τους σε τακτικές γραμμές και στήλες. Για άλλη μια φορά, ο χώρος γύρω από τα κελιά αφήνεται για την καλωδίωση. Για μια λεπτομερή περιγραφή των στυλ διάταξης, δείτε Muroga [1982] και Ueda et al. [1986].

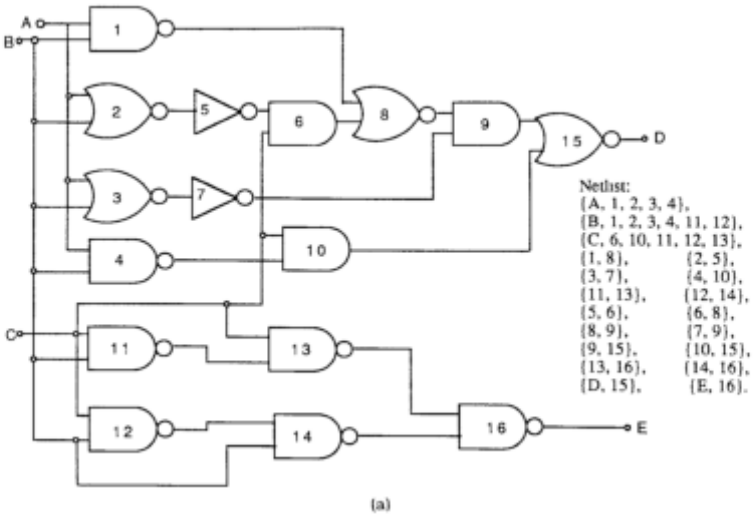
Εικόνα 3 Διάταξη macro block



Το πρόβλημα της τοποθέτησης μπορεί να οριστεί ως εξής. Λαμβάνοντας υπόψη ένα ηλεκτρικό κύκλωμα που αποτελείται από κελιά με προκαθορισμένες εισόδους και εξόδους που αλληλοσυνδέονται με ένα προκαθορισμένο τρόπο, να κατασκευάσετε μια διάταξη που να δείχνει τις θέσεις των κελιών έτσι ώστε το μήκος των καλωδίων και το μέγεθος της διάταξης να είναι τα ελάχιστα δυνατά. Το αρχικό πρόβλημα είναι η περιγραφή των κελιών, αφού αυτά αποτελούνται από διάφορα σχήματα, διάφορα μεγέθη, διαφορετικά terminal locations και netlist, και το πώς θα διαμορφωθούν οι διασυνδέσεις μεταξύ των τερματικών και των κελιών. Η έξοδος είναι κατάλογος των x και y-συντεταγμένων για όλες τις ενότητες. Το Σχήμα 4 μας δίνει ένα παράδειγμα τοποθέτησης, όπου το κύκλωμα του σχήματος 4a τοποθετείται στο standard cell στυλ διάταξης του Σχήματος 4b. Το Σχήμα 4c απεικονίζει το μοντέλο τοποθέτησης Checkerboard στο οποίο όλα τα κελιά υποτίθεται ότι είναι τετράγωνα και ίσου μέγεθος και όλα τα τερματικά υποτίθεται ότι βρίσκονται στο κέντρο των κελιών. Έτσι, το μήκος της σύνδεσης από το ένα κελί στο άλλο είναι μία μονάδα.

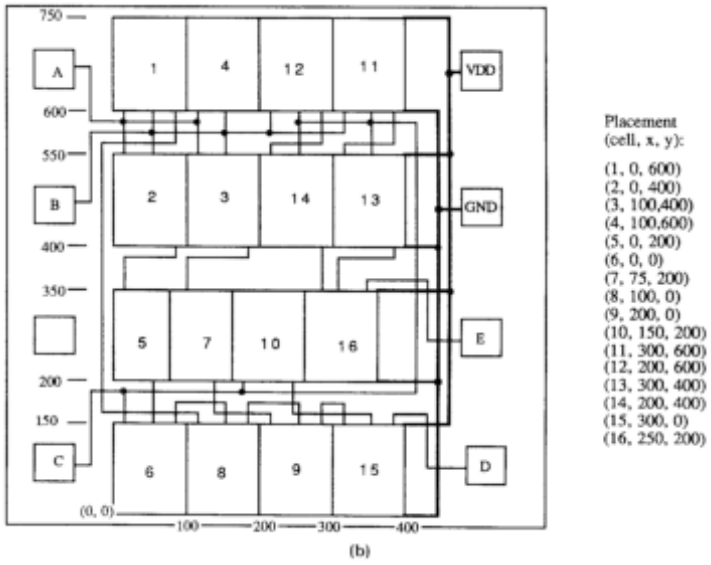
Εικόνα 4 Τοποθέτηση των κελιών: ορισμός του προβλήματος.

(a) είσοδος: netlist



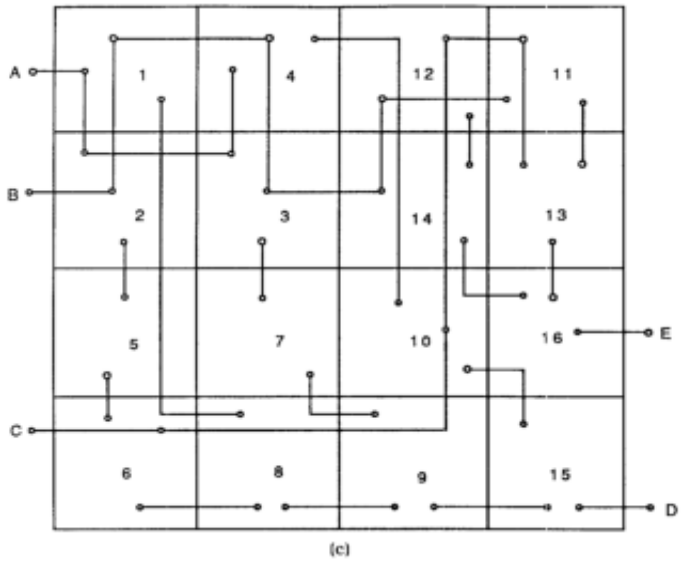
(a)

(b) έξοδος: οι συντεταγμένες της μονάδας



(b)

(c) το μοντέλο checkboard



Οι κύριοι στόχοι ενός αλγορίθμου τοποθέτησης είναι να ελαχιστοποιηθεί το συνολικό εμβαδόν του τσιπ και το συνολικό εκτιμώμενο μήκος των καλωδίων για όλα τα neta (nets). Πρέπει να βελτιστοποιεί την χρήση μιας δεδομένης περιοχή του τσιπ για να χωράει περισσότερη λειτουργικότητα. Πρέπει να ελαχιστοποιηθεί και το μήκος των καλωδίων προκειμένου να μειωθούν οι χωρητικές καθυστερήσεις που σχετίζονται με μεγάλα neta ώστε να επιταχύνουμε τη λειτουργία του τσιπ. Οι στόχοι αυτοί συνδέονται στενά μεταξύ τους για τους τρόπους σχεδιασμού standard cell και gate array, δεδομένου ότι η συνολική περιοχή του τσιπ είναι περίπου ίση με την περιοχή των ενοτήτων(modules) συν την περιοχή που καταλαμβάνεται από την διασύνδεση. Συνεπώς, η ελαχιστοποίηση του μήκος των καλωδίων είναι περίπου ισοδύναμη με την ελαχιστοποίηση της περιοχής του τσιπ. Στο στυλ σχεδιασμού macro, τα ακανόνιστα μεγέθη macros δεν ταιριάζουν πάντα μαζί και κάποιο διάστημα σπαταλιέται. Αυτό διαδραματίζει σημαντικό ρόλο στον καθορισμό της συνολικής περιοχής του τσιπ, και έχουμε ένα trade-off (από κάπου θα χάσουμε) μεταξύ της ελαχιστοποίησης της περιοχής και της ελαχιστοποίησης του μήκους των καλωδίων. Σε ορισμένες περιπτώσεις, μπορεί να χρειαστούν κάποια δευτερεύοντα μέτρα για την απόδοση, όπως η ελαχιστοποίηση του μήκους των καλωδίων σε μερικά κρίσιμα neta, αυξάνοντας, όμως, το συνολικό μήκος του καλωδίου.

Ένα άλλο κριτήριο για μια αποδεκτή τοποθέτηση είναι ότι θα πρέπει να είναι πρακτικά δυνατόν, δηλαδή, (1) οι μονάδες δεν θα πρέπει να επικαλύπτονται, (2) θα πρέπει να βρίσκονται εντός των ορίων του τσιπ, (3) τα standard cells θα πρέπει να περιορίζονται σε σειρές και σε προκαθορισμένες θέσεις, (4) οι πύλες σε ένα gate array θα πρέπει να περιορίζονται σε σημεία του δικτύου. Είναι κοινή πρακτική να ορίσουμε μια συνάρτηση κόστους ή μια αντικειμενική συνάρτηση, η οποία θα αποτελείται από το άθροισμα της συνολικού εκτιμώμενου μήκους καλωδίου, τις διάφορες ποινές αλληλοεπικάλυψης ενοτήτων, την συνολική περιοχή του τσιπ, κ.ο.κ.. Ο στόχος του αλγορίθμου είναι να καθορίσει μια τοποθέτηση με το ελάχιστο δυνατό κόστος.

Ορισμένοι αλγόριθμοι τοποθέτησης που περιγράφονται εδώ, είναι κατάλληλοι για standard cells και gate arrays, μερικοί αλγόριθμοι είναι πιο κατάλληλοι για macroblocks και μερικοί είναι κατάλληλοι και για τα δύο. Στο έγγραφο αυτό, η λέξεις ενότητα (module), κελί (cell), και στοιχείο (element) χρησιμοποιούνται για να περιγράψουν είτε ένα standard cell είτε μια πύλη (ή ένα macroblock, εάν ο αλγόριθμος μπορεί επίσης να χρησιμοποιηθεί για macros). Οι λέξεις macro και block χρησιμοποιούνται ως συνώνυμα στη θέση του macroblock. Η χρήση τους εξαρτάται επίσης από την χρήση τους στις αναφορές. Ομοίως, neto (net), σύρμα (wire), διασύνδεση (interconnect), και η γραμμή σήματος (signal line) χρησιμοποιούνται συνώνυμα. Οι όροι διαμόρφωση, τοποθέτηση, και η λύση (στο πρόβλημα της τοποθέτησης) χρησιμοποιούνται ως συνώνυμα, ώστε να αντιπροσωπεύουν μια ανάθεση των ενοτήτων στις φυσικές τοποθεσίες πάνω στο τσιπ. Η όροι ακροδέκτες (pins) και τερματικό (terminal) αναφέρονται σε τερματικά των modules. Οι ακροδέκτες του τσιπ αναφέρονται ως pads.

Η τοποθέτηση ενοτήτων είναι ένα NP πρόβλημα και, κατά συνέπεια, δεν μπορεί να λυθεί σε πολυωνυμικό χρόνο [1980 Donath; Leighton 1983, Sahni 1980]. Προσπαθώντας να πάρουμε μια μοναδική λύση από την αξιολόγηση της κάθε δυνατής τοποθέτησης ώστε να λάβουμε το καλύτερο δυνατό αποτέλεσμα θα χρειαστεί χρόνο ανάλογο με το παραγοντικό του αριθμού των μονάδων. Αυτή η μέθοδος είναι, συνεπώς, αδύνατο να χρησιμοποιηθεί για κυκλώματα με μεγάλο αριθμό modules. Για να ψάξεις μέσω ενός αριθμού υποψηφίων λύσεων τοποθέτησης αποτελεσματικά, χρειάζεται να χρησιμοποιηθεί ένας ευρετικός αλγόριθμος. Η ποιότητα της επικρατούσας τοποθέτησης εξαρτάται από τον ευρετικό αλγόριθμο. Στην καλύτερη περίπτωση, μπορούμε να ελπίζουμε ότι θα βρούμε μια καλή τοποθέτηση με μήκος σύρματος αρκετά κοντά στο ελάχιστο δυνατό, χωρίς καμία εγγύηση για την επίτευξη του απόλυτα ελαχίστου. Ο στόχος της παρούσας εργασίας είναι να εισαγάγει τον αναγνώστη στους διάφορους αλγορίθμους που έχουν κατασκευαστεί για την επίλυση αυτού του υπολογιστικά δυσεπίλυτου προβλήματος και να αναλύσει την απόδοσή τους.

Η διαδικασία της τοποθέτησης ακολουθείται από αυτή της δρομολόγησης, δηλαδή, τον καθορισμό της διάταξης των διασυνδέσεων μέσω του διαθέσιμου χώρου. Η εύρεση της βέλτιστης δρομολόγησης σε μια τοποθέτηση είναι επίσης, ένα NP πρόβλημα. Πολλοί αλγόριθμοι λειτουργούν επαναληπτικά με τη βελτίωση της τοποθέτησης και, σε κάθε βήμα, υπολογίζουν το μήκος σύρματος μιας ενδιάμεσης διαμόρφωσης. Δεν είναι εφικτό να δρομολογείται κάθε ενδιάμεση διαμόρφωση και να καθορίζεται η ποιότητά της. Αντ' αυτού εκτιμούμε το μήκος του καλωδίου.

Ταξινόμηση των Αλγορίθμων Τοποθέτησης

Οι αλγόριθμοι τοποθέτησης μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες: δομικής τοποθέτησης (constructive placement) και επαναληπτικής βελτίωσης (iterative improvement). Η constructive placement, είναι μια μέθοδος που χρησιμοποιείται για να δημιουργήσει μια τοποθέτηση από την αρχή. Στην iterative improvement, οι αλγόριθμοι ξεκινούν από μια αρχική τοποθέτηση και επαναληπτικά την τροποποιούν σε αναζήτηση μειωμένου κόστους. Εάν μια τροποποίηση οδηγήσει σε μείωση του κόστους, γίνεται αποδεκτή, αλλιώς απορρίπτεται.

Οι αρχικοί αλγόριθμοι constructive placement σε γενικές γραμμές βασιζόντουσαν σε θεμελιώδεις κανόνες συνδεσιμότητας. Για παράδειγμα, δείτε Fukunaga et al. [1983], Hanan [1972a], Kambe et al. [1982], Kang [1983], Kozawa et al. [1983], Magnuson [1977], και ο Persky et al. [1976]. Τυπικά, ένα seed-module επιλέγεται και τοποθετείται στην περιοχή διάταξης του τσιπ. Τότε τα άλλα modules επιλέγονται ένα κάθε φορά ανάλογα με τη σύνδεσή τους στις ήδη υπάρχουσες ενότητες (οι περισσότερο πυκνές συνδέονται πρώτες) και τοποθετούνται σε κενές θέσεις, δίπλα στα modules που έχουν ήδη τοποθετηθεί, έτσι ώστε το μήκος του καλωδίου να είναι το ελάχιστο. Τέτοιοι αλγόριθμοι είναι γενικά πολύ γρήγοροι, αλλά συνήθως οδηγούν σε διατάξεις χαμηλής αξίας. Αυτοί οι αλγόριθμοι χρησιμοποιούνται σήμερα για την παραγωγή μιας αρχικής τοποθέτησης. Ο κύριος λόγος για τη χρήση τους είναι η ταχύτητα τους. Καταναλώνουν μια αμελητέα ποσότητα χρόνου υπολογισμού σε σύγκριση με αλγόριθμους iterative improvement και παρέχουν ένα καλό σημείο εκκίνησης για αυτούς. Ο Palczewski [1984] συζητά την πολυπλοκότητα αυτών των αλγορίθμων. Πιο πρόσφατοι αλγόριθμοι constructive placement, όπως οι αριθμητικές τεχνικές βελτιστοποίησης, τοποθέτησης με διαχωρισμό, και μία force-directed τεχνική που συζητάμε εδώ, αποδίδουν καλύτερη διάταξη, αλλά απαιτούν πολύ περισσότερο χρόνο στην CPU.

Οι αλγόριθμοι iterative improvement συνήθως παράγουν καλές συνθήκες τοποθέτησης, αλλά απαιτούν τεράστιο χρόνο υπολογισμού. Η πιο απλή επαναληπτική στρατηγική βελτίωσης εναλλάσσεται με τυχαία επιλεγμένα ζεύγη ενοτήτων και αποδέχεται την ανταλλαγή αν καταλήγει σε μείωση κόστους [Goto και Kuh 1976, Schweikert 1976]. Ο αλγόριθμος τερματίζεται όταν δεν υπάρχει περαιτέρω βελτίωση μετά από έναν μεγάλο αριθμό δοκιμών. Μια βελτίωση πάνω σε αυτόν τον αλγόριθμο επαναλαμβάνεται με την βελτίωση της επαναληπτικής διαδικασίας και τη βελτίωσή της με το άθροισμα των επαναλήψεων που έγιναν με διαφορετικές αρχικές διαμορφώσεις και με την ελπίδα να επιτευχθεί μια καλή διαμόρφωση σε μια από τις δοκιμές. Επί του παρόντος ο πιο δημοφιλής αλγόριθμος iterative improvement είναι ο simulated annealing, ο γενετικός αλγόριθμος, και κάποιες force-directed τεχνικές τοποθέτησης, τις οποίες συζητούμε λεπτομερώς στις ακόλουθες ενότητες.

Άλλες πιθανές ταξινομήσεις αλγορίθμων για τοποθέτηση είναι οι ντετερμινιστικοί αλγόριθμοι και οι αλγόριθμοι πιθανοτήτων. Αλγόριθμοι που λειτουργούν με βάση τους κανόνες της σταθερής συνδεσιμότητας ή φόρμουλας ή καθορίζουν την τοποθέτηση από την επίλυση εξισώσεων είναι ντετερμινιστικοί και πάντα θα παράγουν το ίδιο αποτέλεσμα για ένα συγκεκριμένο πρόβλημα τοποθέτησης. Οι πιθανοτικοί αλγόριθμοι, αντίθετα, λειτουργούν με τυχαία εξέταση διαμορφώσεων και μπορεί να παράγουν διαφορετικό αποτέλεσμα κάθε φορά που τρέχουν. Οι αλγόριθμοι constructive placement είναι συνήθως ντετερμινιστικοί, ενώ οι αλγόριθμοι iterative improvement είναι συνήθως πιθανοτικοί.

Εκτιμήσεις για το μήκος καλωδίου

Για να κάνουμε μια καλή εκτίμηση του μήκους του καλωδίου, θα πρέπει να εξετάσουμε τον τρόπο με τον οποίο γίνεται η δρομολόγηση στην πραγματικότητα με τη βοήθεια των εργαλείων δρομολόγησης. Σχεδόν όλα τα εργαλεία δρομολόγησης χρησιμοποιούν την γεωμετρία Manhattan, δηλαδή μόνο οι οριζόντιες και κατακόρυφες γραμμές χρησιμοποιούνται για να συνδέσουν δυο σημεία.

Επιπρόσθετα, δύο επίπεδα (layers) χρησιμοποιούνται. Στο ένα layer επιτρέπονται μόνο οριζόντιες γραμμές και στο άλλο layer μόνο κατακόρυφες γραμμές. Η συντομότερη διαδρομή για τη σύνδεση ενός σετ ακροδεκτών μαζί, είναι η Steiner tree (Σχήμα 5α). Σε αυτή τη μέθοδο, ένα σύρμα μπορεί να διακλαδίζεται σε οποιοδήποτε σημείο κατά μήκος του. Η μέθοδος αυτή συνήθως δεν χρησιμοποιείται από τους δρομολογητές, λόγω της πολυπλοκότητας του υπολογισμού τόσο του βέλτιστου σημείου διακλάδωσης, όσο και της προκύπτουσας βέλτιστης διαδρομής από το σημείο διακλάδωσης για τους ακροδέκτες. Αντ' αυτού, οι ελάχιστες συνδέσεις του spanning tree και οι συνδέσεις αλυσίδας είναι οι πιο συχνά χρησιμοποιημένες τεχνικές σύνδεσης. Για αλγορίθμους που υπολογίζουν το δέντρο Steiner δείτε Chang [1972], Chen [1983], και Hwang [1976,1979].

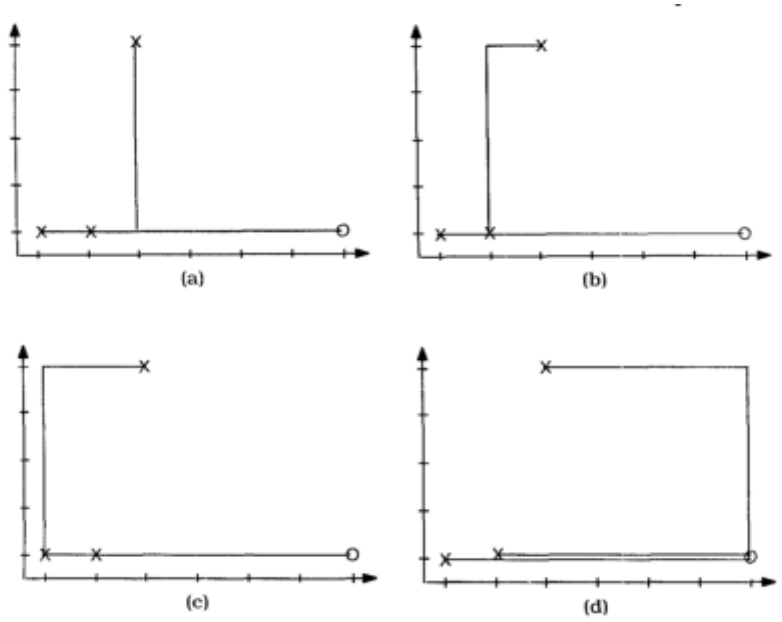
Οι ελάχιστες συνδέσεις του spanning tree (Σχήμα 5β), επιτρέπουν διακλάδωση μόνο στις θέσεις των ακροδεκτών. Άρα, οι ακροδέκτες συνδέονται σχηματίζοντας το ελάχιστο spanning tree. Οι αλγόριθμοι υπάρχουν για την παραγωγή του ελάχιστου spanning tree, δεδομένης της netlist και των συντεταγμένων του κάθε κελιού. Ένα παράδειγμα ελάχιστου spanning tree είναι και ο αλγόριθμος του Kruskal [1956].

Οι αλυσιδωτές συνδέσεις (Σχήμα 5γ) δεν επιτρέπουν καμία διακλάδωση. Κάθε ακρεοδέκτης συνδέεται με τον επόμενο, σχηματίζοντας μια αλυσίδα. Αυτές οι συνδέσεις είναι πιο εύκολο να εφαρμοστούν σε σχέση με τις συνδέσεις του spanning tree, αλλά καταλήγουν σε λίγο μακρύτερες διασυνδέσεις.

Οι συνδέσεις source-to-sink (Σχήμα 5d), όπου η έξοδος ενός module συνδέεται όλες τις εισόδους με ξεχωριστά καλώδια, είναι η πιο απλή να εφαρμοστεί. Μπορούν, ωστόσο, να καταλήγουν σε υπερβολικό μήκος διασύνδεσης και σε σημαντική συμφόρηση καλωδίων. Ως εκ τούτου, αυτό το είδος σύνδεσης χρησιμοποιείται σπάνια.

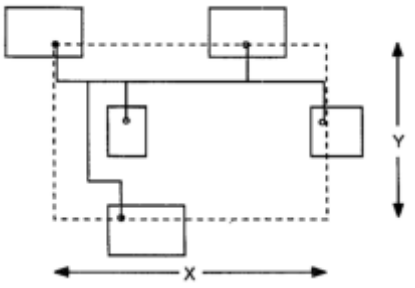
Μια αποδοτική και συχνά χρησιμοποιούμενη μέθοδος για την εκτίμηση του μήκους καλωδίου είναι η μέθοδος semiperimeter. Το μήκος του καλωδίου προσεγγίζεται από το μισό της περιμέτρου, του μικρότερου ορθογωνίου οριοθέτησης που περικλείει όλα τα pins (Σχήμα 6). Για την καλωδίωση «Μανχάταν», η μέθοδος αυτή δίνει το ακριβές μήκος καλωδίου για όλα τα neta two-terminals και three-terminals, υπό τον όρο ότι κατά τη δρομολόγηση δεν υπάρχει υπέρβαση της οριοθέτησης του ορθογωνίου. Για τα neta four-terminals, στη χειρότερη περίπτωση, η εκτίμηση της μεθόδου semiperimeter προβλέπει καλώδιο μήκους 33% μικρότερο τόσο από την πραγματική αλυσίδα σύνδεσης όσο και από το αποτέλεσμα του spanning tree. Για neta με περισσότερα pins και με περισσότερες συνδέσεις «ζιγκ-ζαγκ», το μήκος του καλωδίου που υπολογίζει ο semiperimeter είναι γενικά μικρότερο από το πραγματικό μήκος του καλωδίου. Άλλωστε, αυτή η μέθοδος παρέχει την καλύτερη εκτίμηση για το πιο αποτελεσματικό σύστημα καλωδίωσης, το δέντρο Steiner (Steiner tree). Το σφάλμα θα είναι μεγαλύτερο για minimal spanning tree και ακόμα μεγαλύτερο για συνδέσεις αλυσίδων. Στην πράξη, πάντως, τα κυκλώματα two και three-terminals είναι πιο κοινά. Ακόμη, για τα πιο πολύπλοκα neta, το μήκος του καλωδίου semiperimeter είναι μια καλή εκτίμηση, καθως δεν θα βρίσκονται όλα στη χειρότερη περίπτωση της μεθόδου.

Εικόνα 5 Σχήματα καλωδίωσης. (a) Steiner tree – μήκος καλωδίου = 10. (b) το ελάχιστο spanning tree – μήκος καλωδίου = 11. (c) Οι αλυσιδωτές συνδέσεις – μήκος καλωδίου = 12 (d) Οι συνδέσεις source-to-sink – μήκος καλωδίου = 19. Ο = η πηγή, Χ = η καταβόθρα



Μερικοί από τους αλγορίθμους που περιγράφονται στη παράγραφο 4 χρησιμοποιούν το ευκλείδειο μήκος καλωδίου ή το τετράγωνο του ευκλείδειου μήκους καλωδίου. Το τετραγωνισμένο μήκος καλωδίου χρησιμοποιείται για να ελλατώσουμε το χρόνο που απαιτείται για τον υπολογισμό μιας τετραγωνικής ρίζας και για υπολογισμούς κινητής υποδιαστολής σε σύγκριση με επεξεργασία ακεραίων. Βελτιστοποίηση του τετραγωνισμένου μήκους σύρματος θα διασφαλίσει ότι το ευκλείδειο μήκος του καλωδίου έχει βελτιστοποιηθεί.

Εικόνα 6 Μήκος καλωδίου με semiperimeter



1. Simulated Annealing

1.1 Αλγόριθμος

Ο Simulated Annealing [Kirkpatrick et al. 1983] είναι κατά πάσα πιθανότητα η πιο καλά ανεπτυγμένη μέθοδος που είναι διαθέσιμη σήμερα για την τοποθέτηση των modules. Είναι πολύ χρονοβόρα αλλά δίνει άριστα αποτελέσματα. Είναι εξαιρετική για την επίλυση κάθε συνδυαστικού προβλήματος βελτιστοποίησης, όπως το πρόβλημα του Traveling Salesman [Randelman και Grest 1986] ή για προβλήματα VLSI-CAD, το PLA folding [Wong et al. 1986], διαχωρισμού [Chung και Rao 1986], δρομολόγησης [Vecchi και Kirkpatrick 1983], λογική ελαχιστοποίηση [Lam και Delosme 1986], floor planning [Otten και van Ginnekin 1984], ή τοποθέτησης. Μπορεί να θεωρηθεί μία βελτιωμένη έκδοση του αλγόριθμου της απλής τυχαίας ανταλλαγής ζευγών. Αυτός ο τελευταίος αλγόριθμος έχει την τάση να κολλάει σε τοπικά ελάχιστα. Ας υποθέσουμε, για παράδειγμα, κατά την εκτέλεση του αλγορίθμου ανταλλαγής ζεύγων, συναντάμε μια διαμόρφωση που έχει πολύ μεγαλύτερο κόστος από το βέλτιστο και καμία ανταλλαγή ζευγών δεν μπορεί να μειώσει το κόστος. Δεδομένου ότι ο αλγόριθμος δέχεται εναλλαγές μόνον εάν υπάρχει μια μείωση του κόστους και δεδομένου ότι εξετάζει μόνο ζεύγη κόμβων, δεν υπάρχει τρόπος περαιτέρω βελτίωσης από την υπάρχουσα διαμόρφωση. Ο αλγόριθμος είναι παγιδευμένος σε μια τοπική βέλτιστη διαμόρφωση, η οποία μπορεί να είναι φτωχή. Η πείρα δείχνει ότι αυτό συμβαίνει αρκετά συχνά. Για να αποφευχθούν τέτοιου είδους τοπικά βέλτιστα χρειαζόμαστε έναν αλγόριθμο που περιοδικά δέχεται κινήσεις που έχουν ως αποτέλεσμα μια αύξηση του κόστους. Το simulated annealing κάνει ακριβώς αυτό.

1.1 Αλγόριθμος

A typical simulated annealing algorithm is as follows:

```
PROCEDURE Simulated_Annealing;
    initialize;
    generate random configuration;
    WHILE stopping_criterion (loop_count, temperature) = FALSE
        WHILE inner_loop_criterion = FALSE
            new_configuration ← perturb(configuration);
            ΔC ← evaluate(new_configuration, configuration);
            IF ΔC < 0 THEN new_configuration ← configuration
            ELSE IF accept(ΔC, temperature) > random(0, 1)
                    THEN new_configuration ← configuration;
                    ENDIF
            ENDIF
        ENDWHILE
        temperature ← schedule(loop_count, temperature);
        loop_count ← loop_count + 1;
    ENDWHILE
END.
```

Η βασική διαδικασία στο simulated annealing είναι να δέχεται όλες τις κινήσεις που έχουν ως αποτέλεσμα τη μείωση του κόστους. Οι κινήσεις που έχουν ως αποτέλεσμα

την αύξηση του κόστους είναι αποδεκτές με την πιθανότητα της μείωσης όσο αυξάνεται το κόστος. Μια παράμετρος T, που ονομάζεται θερμοκρασία (temperature), χρησιμοποιείται για να ελέγχει την πιθανότητα αποδοχής κινησεων που αυξάνουν το κόστος. Όσο υψηλότερο είναι το T, τόσες περισσότερες τέτοιες κινήσεις γίνονται αποδεκτές. Στις περισσότερες υλοποιήσεις του παρόντος αλγορίθμου, η πιθανότητα αποδοχής δίνεται από το τύπο exp(-ΔC / T), όπου ΔC είναι η αύξηση του κόστους. Στην αρχή, το temperature έχει ρυθμιστεί σε πολύ υψηλή τιμή, ώστε οι περισσότερες από τις κινήσεις να γίνονται δεκτές. Στη συνέχεια, η θερμοκρασία μειώνεται σταδιακά, ώστε οι κινήσεις αύξησης του κόστους να έχουν λιγότερες πιθανότητες να γίνουν αποδεκτές. Τελικά, η θερμοκρασία μειώνεται σε μια πολύ χαμηλή τιμή, ώστε να γίνονται δεκτές μόνο κινήσεις που προκαλούν τη μείωση του κόστους, και ο αλγόριθμος συγκλίνει σε ένα χαμηλό κόστος διαμόρφωσης.

Το Perturb δημιουργεί μια τυχαία εκδοχή της τρέχουσας διαμόρφωσης. Αυτό μπορεί να περιλαμβάνει μετακίνηση ενός module σε τυχαία θέση, αμοιβαία ανταλλαγή θέσων δύο modules, περιστροφή και κατοπτρισμό εντός των ορίων που επιτρέπει η γεωμετρία της διάταξης, ή οποιαδήποτε άλλη κίνηση μπορεί να μεταβάλει το μήκος του καλωδίου. Για τα standard cells συνήθως επιτρέπεται ο κατοπτρισμός περί τον κατακόρυφο άξονα, ενώ για τα macro block επιτρέπεται περιστροφή 90 μοιρών ή κατοπτρισμός γύρω και από τους δυο άξονες. Μια συνάρτηση με range-limiting μπορεί να εφαρμοστεί, η οποία μπορεί αρχικά να επιλέξει το module να μετακινηθεί, και στη συνέχεια να επιλέξει έναν προορισμό μέσα σε ένα καθορισμένο εύρος από τη θέση προορισμού για αυτό. Αυτό γίνεται συνήθως για να αυξηθεί το ποσοστό αποδοχής των κινήσεων.

Η *Αξιολόγηση* αξιολογεί την αλλαγή στο κόστος, χρησιμοποιώντας τη μέθοδο semiperimeter. Για να εξοικονομήσουμε χρόνο στην CPU, η αλλαγή στο μήκος του καλωδίου μπορεί να υπολογιστεί σταδιακά. Δηλαδή, ο υπολογισμός γίνεται μόνο για τα neta που συνδέονται με τα cells που έχουν μετακινηθεί.

Η *Αποδοχή* είναι η πιθανοτική συνάρτηση αποδοχής που καλείται όταν το κόστος αυξάνεται από μια σύγχυση .Καθορίζει είτε να αποδεχθεί μια κίνηση ή όχι, ανάλογα με την αύξηση του κόστους και τη θερμοκρασία. Συνήθως είναι η εκθετική συνάρτηση που περιγράφεται παραπάνω. αλλά μπορεί να είναι και οποιαδήποτε άλλη συνάρτηση.

Το Schedule, είναι το πρόγραμμα της μεταβλητής Temperature, το οποίο δίνει την επόμενη ως συνάρτηση του αριθμού των επαναλήψεων ή των προηγούμενων τιμών. Για παράδειγμα, η συνάρτηση $\boldsymbol{T_{i+1} = 0.1T_i}$, μπορεί να χρησιμοποιηθεί για εκθετική μείωση του temperature.

Το Inner_loop_criterion είναι το κριτήριο που αποφασίζει για τον αριθμό των δοκιμών σε κάθε temperature. Συνήθως ο αριθμός των κινήσεων που επιχειρούν ανά κελί σε κάθε τιμή του temperature είναι δεδομένος.

Το *Stopping_criterion* τερματίζει τον αλγόριθμο όταν το temperature ή ο αριθμός των επαναλήψεων έχει φθάσει μία κατώτατη τιμή.

Δεν υπάρχουν σταθεροί κανόνες για την αρχική θερμοκρασία, το ψυκτικό χρονοδιάγραμμα, η πιθανοτική συνάρτηση αποδοχής, ή το stopping criterion, ούτε

υπάρχουν οποιοιδήποτε περιορισμοί για το τύπο κινήσεων που θα χρησιμοποιηθούν -μετατόπιση, ανταλλαγή, περιστροφή, και ούτω καθεξής. Η ποιότητα της τοποθέτησης και του χρόνου εκτέλεσης εξαρτάται από αυτές τις παραμέτρους. Μια καλή επιλογή των παραμέτρων μπορεί να οδηγήσει σε μία καλή τοποθέτηση σε σχετικά σύντομο χρονικό διάστημα λειτουργίας. Η μεγαλύτερη πρόκληση του αλγόριθμου simulated annealing έγκειται στην εύρεση ενός ενιαίου συνόλου των παραμέτρων και συναρτήσεων που δίνουν ,σταθερά ,πολύ καλές λύσεις για μια μεγάλη ποικιλία κυκλωμάτων, χρησιμοποιώντας παράλληλα ελάχιστο χρόνο υπολογισμού. Αρχικά, οι ερευνητές επιλέξανε αυτές τις παραμέτρους και συναρτήσεις αυθαίρετα. Πρόσφατα, ωστόσο, αρκετοί ερευνητές έχουν κάνει μια αυστηρή στατιστική ανάλυση του simulated annealing προκειμένου να αντλήσουν πιο κατάλληλες συναρτήσεις. Η Ενότητα 1.3 δίνει τις παραμέτρους και λειτουργίες που χρησιμοποιούνται στον Timberwolf, ένα γνωστό πακέτο τοποθέτησης και δρομολόγησης . Η Ενότητα 1.4 ασχολείται με άλλες εναλλακτικές λύσεις για αυτές τις παραμέτρους και λειτουργίες.

1.2 Λειτουργία του simulated annealing.

Εάν το simulated annealing εκτελείται για ένα μεγάλο χρονικό διάστημα και με το κατάλληλο χρονοδιάγραμμα ψύξης, είναι εγγυημένο ότι θα συγκλίνει σε ένα ολικό ελάχιστο [Mitraetal, 1985, van Laarhoven και Aarts1987]. Αυτή η ενότητα εξηγεί γιατί αυτό ισχύει. Δύο είναι οι αναλογίες που δίδονται για να επεξηγήσουν την λειτουργία αυτού του αλγορίθμου.

Στην πρώτη αναλογία, από την οποία ο αλγόριθμος παίρνει το όνομά του, το simulated annealing συγκρίνεται με την διεργασία τοποθέτησης σε μέταλλα. Εάν ένα μέταλλο έχει παραμορφωθεί και έχει ατελή κρυσταλλική δομή, ένας τρόπος για την αποκατάσταση της ατομική τοποθέτησης είναι να θερμανθεί σε πολύ υψηλή θερμοκρασία, και στη συνέχεια να ψυχθεί πολύ αργά. Σε υψηλή θερμοκρασία, τα άτομα έχουν επαρκή κινητική ενέργεια για να σπάσουν από τις τρέχουσες εσφαλμένες τους θέσεις. Καθώς το υλικό δροσίζει, τα άτομα σιγά-σιγά παγιδεύονται και αρχίζουν να παίρνουν τις σωστές θέσεις στο πλέγμα. Εάν το υλικό ψύχεται πολύ γρήγορα, τα άτομα δεν έχουν την ευκαιρία να πάρουν τις σωστές συντεταγμένες στο πλέγμα, και έτσι καταψύχονται ελαττωματικά εντός της δομής του κρυστάλλου. Ομοίως, στη simulated annealing σε υψηλή θερμοκρασία, υπάρχουν πολλές τυχαίες παραλλαγές στην αρχική διαμόρφωση. Αυτό δίνει στα κελιά που είναι σε εσφαλμένη περιοχή την ευκαιρία να αποσπασθούν από την αρχική τους θέση. Καθώς η θερμοκρασία μειώνεται, τα κελιά αρχίζουν σιγά-σιγά να παγιδεύονται σε βέλτιστες θέσεις .

Στη δεύτερη αναλογία, η δράση του simulated annealing συγκρίνεται με μια μπάλα σε μια λοφώδη έκταση, μέσα σε ένα κουτί [Szu 1986]. Χωρίς οποιαδήποτε διαταραχή, η μπάλα θα κυλήσει στο λόφο μέχρι να βρει ένα λάκκο, όπου θα σταματήσει για πάντα παρόλο που ο λάκκος μπορεί να είναι ψηλά και όχι κάτω-κάτω στη κοιλάδα. Για να έχουμε τη μπάλα κάτω-κάτω στη κοιλάδα, το πλαίσιο πρέπει να κουνηθεί έντονα και αρκετά έτσι ώστε η μπάλα να διασχίσει την ψηλότερη κορυφή στο δρόμο του. Κατά την ίδια στιγμή, πρέπει να κουνηθεί ελαφρά έτσι ώστε όταν η μπάλα φτάσει στο κατώτερο σημείο του λόφου να μην μπορεί να βγει. Πρέπει επίσης να κουνηθεί για αρκετά μακρύ διάστημα έτσι ώστε να υπάρχει υψηλή πιθανότητα να φτάσει στο

κατώτερο σημείο του λόφου. Αυτά τα χαρακτηριστικά μεταφράζονται άμεσα σε παραμέτρους αλγορίθμων. Η δύναμη ή η πραότητα των δονήσεων καθορίζεται από την πιθανοτική συνάρτηση αποδοχής και την αρχική θερμοκρασία, και η διάρκεια των δονήσεων εξαρτάται από το ψυκτικό χρονοδιάγραμμα και το κριτήριο εσωτερικού βρόχου.

1.3 Timberwolf 3.2

Το Timberwolf, που αναπτύχθηκε από τον Carl Sechen και τον Sangiovanni-Vincentelli είναι ένα ευρέως χρησιμοποιούμενο και ιδιαίτερα επιτυχημένο πακέτο για τοποθέτηση και δρομολόγηση βασισμένο στο simulated annealing. Διαφορετικές εκδόσεις του Timberwolf έχουν αναπτυχθεί για τη τοποθέτηση standard cells [Sechen 1986, 1988b, Sechen και Sangiovanni-Vincentelli 1986], macros [Cassoto et al. 1987], και για το floor planning [Sechen 1988a]. Η έκδοση 3.2 για τα standard cells θα περιγραφεί εδώ.

Το Timberwolf κάνει την τοποθέτηση και τη δρομολόγηση σε τρία στάδια. Στο πρώτο στάδιο, τα κελιά είναι τοποθετημένα έτσι ώστε να ελαχιστοποιείται το εκτιμώμενο μήκος καλωδίου χρησιμοποιώντας simulated annealing. Στο δεύτερο στάδιο, τα κελιά εισάγονται, όπως απαιτείται, με το μήκος του καλωδίου να είναι το ελάχιστο δυνατό και πάλι, και προκαταρκτικά γίνεται global δρομολόγηση. Στο τρίτο στάδιο, γίνονται τοπικές αλλαγές στην τοποθέτηση οπουδήποτε είναι δυνατόν να μειωθεί ο αριθμός των κομματιών καλωδίου που απαιτούνται για καλωδίωση. Στη συζήτηση που ακολουθεί θα ασχοληθούμε κατά κύριο λόγο με το στάδιο 1-τοποθέτηση. Λεπτομέρειες για το υπόλοιπο του αλγορίθμου δίνεται στον Sechen [1986, 1988b] και Sechen και Sangiovanni-Vincentelli [1986].

Οι παράμετροι που χρησιμοποιούνται για το simulated annealingtou Timberwolf έχουν ως εξής.

1.3.1 Συνάρτηση Move Generation

Δύο μέθοδοι χρησιμοποιούνται για τη δημιουργία νέων συνθέσεων από την τρέχουσα διαμόρφωση. Είτε ένα κελί επιλέγεται τυχαία και μετατοπίζεται σε μια τυχαία θέση στο τσιπ, ή δύο κελιά επιλέγονται τυχαίως και εναλλάσσονται. Η απόδοση του αλγορίθμου παρατηρήθηκε να εξαρτάται από το r, όπου r ο λόγος των μετατοπίσεων σε κόμβους. Πειραματικά αποτελέσματα που δίνονται στο Sechen και Sangiovanni-Vincentelli [1986] αναφέρουν ότι ο αλγόριθμος έχει την καλύτερη επίδοση όταν $3 < r < 8$.

Κατοπτρισμός κελιού περί τον οριζόντιο άξονα γίνεται επίσης, αλλά μόνο όταν μία μετατόπιση απορρίπτεται και μόνο σε περίπου 10% αυτών των περιπτώσεων επιλέγεται τυχαία. Επιπλέον, μία θερμοκρασία εξαρτώμενη από το περιορισμό εύρους χρησιμοποιείται για να περιορίσει την απόσταση επί της οποίας ένα κελί μπορεί να κινηθεί. Αρχικά, το εύρος της περιορισμένης εμβέλειας είναι διπλάσιο από το εύρος του τσιπ, έτσι ώστε για μια σειρά από υψηλές θερμοκρασίες δεν έχουμε κανένα περιορισμό. Το εύρος μειώνεται λογαριθμικά με τη θερμοκρασία:

$$L_{WV}(T) = L_{WV}(T_1)\frac{\log T}{\log T_1}$$

$$L_{WH}(T) = L_{WH}(T_1)\frac{\log T}{\log T_1}$$

όπου $L_{WV}(T_1)$ και $L_{WH}(T_1)$ είναι η επιθυμητές αρχικές τιμές του κατακόρυφου και οριζόντιου εύρος παραθύρου $L_{WV}(T)$ και $L_{WH}(T)$, αντίστοιχα.

1.3.2 Cost Function

Η συνάρτηση κόστους είναι το άθροισμα τριών συστατικών: το κόστος μήκος του καλωδίου, C_1, η ποινή επικάλυψης των στοιχείων, C_2, και η ποινή του μήκος της γραμμής ελέγχου, C_3.

Το μήκος του κόστους καλωδίου C_1 υπολογίζεται χρησιμοποιώντας τη μέθοδο semiperirneter, με στάθμιση των κρίσιμων δικτύων και ανεξάρτητη στάθμιση της οριζόντιας και κάθετης καλωδίωσης για κάθε neto:

$$C_1 = \sum_{\text{nets}} \left[x(i)W_H(i) + y(i)W_V(i) \right],$$

όπου x(i) και y(i) είναι τα κατακόρυφα και οριζόντια ζεύγη του ορθογωνίου δικτύου οριοθέτησης, και $W_H(i)$ και $W_V(i)$ είναι τα βάρη των οριζόντιων και κάθετων ζευγών καλωδίωσης. Κρίσιμα neta είναι εκείνα που πρέπει να βελτιστοποιηθούν περισσότερο από τα υπόλοιπα, ή αυτά που πρέπει να περιορίζεται σε ένα ορισμένο μέγιστο μήκος λόγω της καθυστέρησης διάδοσης. Εάν έχουν τοποθετηθεί σε μεγαλύτερο βάρος, ο αλγόριθμος θα τοποθετήσει κατά προτίμηση τα κελιά που συνδέονται με τα κρίσιμα neta κοντά το ένα στο άλλο σε μια προσπάθεια να μειώσει το κόστος. Εάν τα neta εξακολουθούν να υπερβαίνουν το μέγιστο μήκος στην τελική τοποθέτηση, το βάρος τους μπορεί να αυξηθεί και ο αλγόριθμος τρέχει ξανά.

Ανεξάρτητα οριζόντια και κατακόρυφα βάρη δίνουν στο χρήστη την ευελιξία να επιλέξει συνδέσεις προς μία κατεύθυνση σε σχέση με άλλη κατεύθυνση. Έτσι, στη τεχνολογία διπλού μετάλλου, όπου είναι δυνατόν να στοιβαχθούν και να τροφοδοτηθούν στην κορυφή των κελιών και δεν καταλαμβάνουν κανένα επιπλέον χώρο, σε κάθετα ζεύγη μπορεί να δοθεί προτίμηση (μικρότερο βάρος). Κατά την φάση της δρομολόγησης, αυτά τα κελιά συνδέονται με κατευθείαν τροφοδοσία αντί για οριζόντια ζεύγη καλωδίωσης μέσα από κανάλια, και έτσι πολύτιμοι χώροι καναλιών διατηρούνται ελεύθερα. Από την άλλη πλευρά, σε τσιπ, όπου η κατευθείαν τροφοδοσία είναι δαπανηρή από την άποψη της περιοχής, προτιμάτε τα οριζόντια καλωδίωση και τα οριζόντια ζεύγη δικτύων να λαμβάνουν χαμηλότερο βάρος. Αυτό ελαχιστοποιεί τον αριθμό απευθείας τροφοδοσιών που απαιτούνται.

Η ποινή επικάλυψης στοιχείων, C_2, είναι παραβολική στην ποσότητα της επικάλυψης:

$$C_2 = W_2 \sum_{i \neq j} [O(i,j)]^2,$$

όπου το O (i, j) είναι η επικάλυψη μεταξύ της i και j-οστού κελιού, και το W_2 είναι το βάρος αυτής της ποινής. Παρατηρήθηκε ότι το C_2 συγκλίνει στο 0 για W_2 = 1. Η παραβολική συνάρτηση προκαλεί μεγάλες επικαλύψεις που πρέπει να τιμωρηθούν και, ως εκ τούτου τις αποφεύγουμε περισσότερο από τις μικρές. Μολονότι η επικάλυψη κελιών δεν επιτρέπεται στην τελική τοποθέτηση και πρέπει να αφαιρεθεί με μια ελάχιστη μετατόπιση κελιών, χρησιμοποιεί αρκετό χρόνο υπολογισμού για να αφαιρέσει την επικάλυψη για κάθε προτεινόμενη κίνηση. Υπενθυμίζουμε ότι το μήκος καλωδίου υπολογίζεται σταδιακά. Εάν πάρα πολλά κελιά μετατοπιστούν σε μια προσπάθεια να αφαιρέσουμε την επικάλυψη, θα πάρει πάρα πολύ χρόνο υπολογισμού για να προσδιοριστεί η μεταβολή στο μήκος του σύρματος. Για αυτό το λόγο οι αλγόριθμοι επιτρέπουν την επικάλυψη κατά τη διάρκεια του annealing από το να έχουμε penalize. Η επικάλυψη προκαλεί μόνο ένα μικρό σφάλμα στο εκτιμώμενο μήκος καλωδίου. Όσο η επικάλυψη είναι μικρή, αυτό το σφάλμα θα είναι μικρό. Επιπλέον, οι μικρές επικαλύψεις τείνουν να εξουδετερώνονται σε αρκετές επαναλήψεις. Έτσι, είναι πιο σωστό να κάνουμε penalize σε μεγάλες επικαλύψεις πιο δυνατά από ό,τι κάνουμε στις μικρές επικαλύψεις με τη χρήση μιας τετραγωνικής συνάρτησης.

Η ποινή του μήκος της γραμμής ελέγχου, C_3, είναι μια συνάρτηση της διαφοράς μεταξύ του πραγματικού μήκους γραμμής και του επιθυμητού μήκους γραμμής. Τείνει να εξισώσει τα μήκη γραμμών αυξάνοντας το κόστος, αν οι σειρές έχουν άνισα μήκη. Τα άνισα μήκη γραμμής οδηγούν σε σπατάλη χώρου, όπως φαίνεται στο Σχήμα 1a. Η ποινή δίνεται από τη σχέση

$$C_3 = W_3 \sum_{\text{rows}} | L_R - \hat{L}_R |,$$

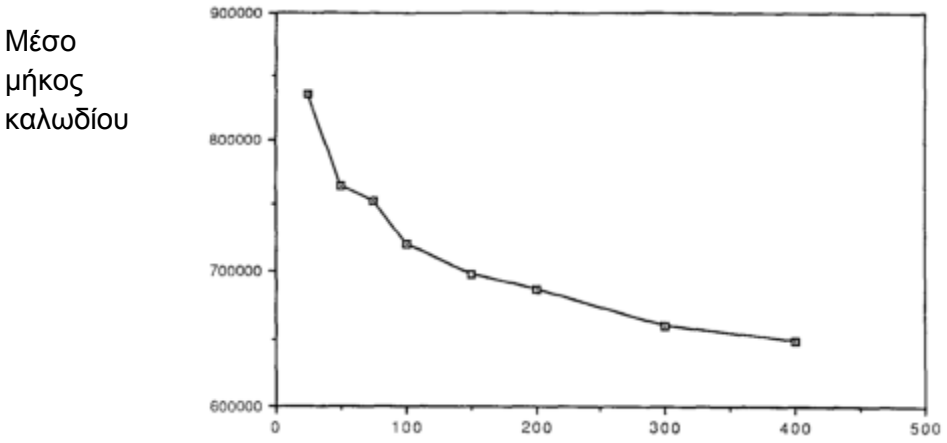
όπου L_R είναι το πραγματικό μήκος γραμμής, $\hat{L}_R$ είναι το επιθυμητό μήκος γραμμής, και W_3 είναι το βάρος αυτής της ποινής για την οποία έχουμε προεπιλεγμένη τιμή το 5. Πειράματα δείχνουν ότι η χρησιμοποιούμενη συνάρτηση παρέχει καλό έλεγχο, με τελικά μήκη γραμμών εντός 3-5% της επιθυμητής τιμής. Αποτελέσματα από δύο πειράματα που δίδονται από τους Sechen και Sangiovanni-Vincentelli [1986], δείχνουν μια μείωση στο μήκος σύρματος όταν εισήχθη η ποινή για το μήκος της γραμμής έλεγχου.

1.3.3 Κριτήριο εσωτερικού Loop

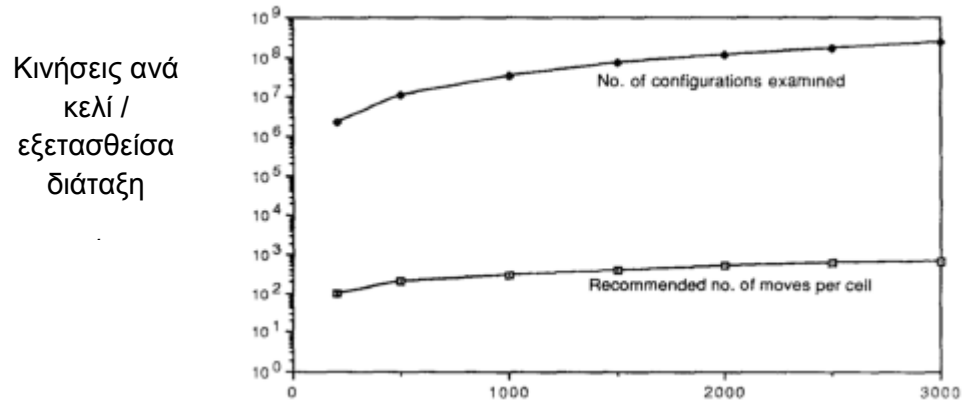
Σε κάθε θερμοκρασία, ένας σταθερός αριθμός κινήσεων ανά κελί επιχειρείται. Αυτός ο αριθμός είναι μια παράμετρος που καθορίζεται από το χρήστη. Το τελικό μήκος σύρματος ελαττώνεται μονοτονικά καθώς ο αριθμός κινήσεων ανά κελί αυξάνεται. Καθώς ο αριθμός των κινήσεων αυξάνεται, η μείωση στο τελικό μήκος του καλωδίου ελαττώνεται , και παρατειρούνται μεγάλες αυξήσεις στο χρόνο CPU. Ο βέλτιστος αριθμός κινήσεων ανά κελί εξαρτάται από το μέγεθος του κυκλώματος. Για παράδειγμα, για ένα κύκλωμα 200 κελιών,100 κινήσεις ανά κελί συνιστόνται σύμφωνα με τον Sechen [1986]. Για ένα κύκλωμα 3000 κελιών, 700 κινήσεις ανά κελί

συνιστόνται, οι οποίες μεταφράζονται σε ένα σύνολο 245.7 εκατομμύρια απόπειρωνς. Το Σχήμα 7a δείχνει το τελικό μήκος του σύρματος ως συνάρτηση του αριθμού των κινήσεων ανά κελί για ένα πρόβλημα 1500 κελιών. Καθώς ο αριθμός των κινήσεων ανά κελί αυξάνεται πέρα από το συνιστώμενο σημείο, η καμπύλη επιπεδώνεται προς τα έξω, προκαλώντας έτσι μικρή περαιτέρω βελτίωση αλλά με τεράστιες αυξήσεις στον χρόνο υπολογισμού. Το Σχήμα 7β δείχνει τη συνιστώσα του αριθμού κινήσεων ανά κελί ως συνάρτηση του μεγέθους του προβλήματος.

Εικόνα 7 (a) Εναλλαγή μεταξύ ποιότητας και χρόνου CPU στο Timber-Wolf



Εικόνα 7 (b) Προτεινόμενος αριθμός κινήσεων ανά κελί



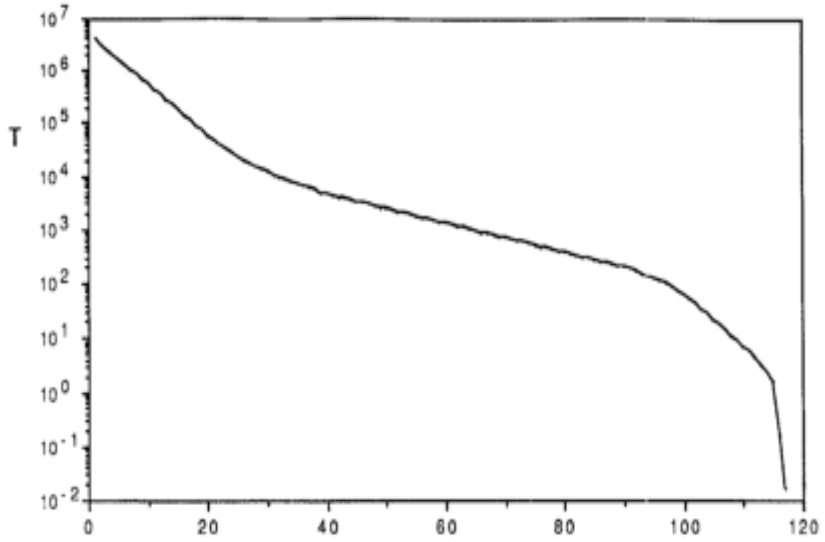
1.3.4 Χρονοδιάγραμμα ψύξης και κριτήριο διακοπής

Το χρονοδιάγραμμα ψύξεως μπορεί να εξηγηθεί από μια αναλογία με τη διαδικασία της κρυστάλλωσης. Για να επιτευχθεί μια τέλεια δομή κρυστάλλου, είναι σημαντικό γύρω από το σημείο τήξεως, η θερμοκρασία να μειώνεται με πολύ βραδύ ρυθμό. Η annealing διαδικασία ξεκίνα σε μια πολύ υψηλή θερμοκρασία, T_1 = 4.000.000, έτσι ώστε οι περισσότερες από τις κινήσεις να γίνονται δεκτές. Το ψυκτικό χρονοδιάγραμμα αντιπροσωπεύεται από την εξίσωση

$$T_{i+1} = \alpha(T)T_i,$$

όπου α(Τ) είναι παράμετρος ρυθμού ψύξης, η οποία καθορίζεται πειραματικά. Στην αρχή, η θερμοκρασία μειώνεται ραγδαία [α(Τ) ≈ 0,8]. Στη συνέχεια, στο μέσο εύρος θερμοκρασίας, η θερμοκρασία μειώνεται βραδέως [α(Τ) ≈ 0.95]. Η μεγαλύτερη επεξεργασία γίνεται σε αυτό το εύρος. Στην κλίμακα χαμηλής θερμοκρασίας, η θερμοκρασία μειώνεται και πάλι ταχέως [α(Τ) ≈ 0,8]. Το προκύπτον χρονοδιάγραμμα ψύξεως φαίνεται στο Σχήμα 8. Ο αλγόριθμος τερματίζει όταν ***T <0.1***. Αυτό αποτελείται από 117 στάδια θερμοκρασίας.

Εικόνα 8 Timber-Wolf 3.2 χρονοδιάγραμμα ψύξης



1.3.5 Απόδοση

Το Σχήμα 9 δείχνει μια τυπική καμπύλη βελτιστοποίησης. Κατά τις πρώτες επαναλήψεις υπάρχει τόσο πολύ τυχαία διατάραξη με το κόστος να αυξάνεται. Κατά το πρώτο ήμισυ της διαδρομής, δεν υπάρχει καμία βελτίωση. Αυτή η διαταραχή είναι απαραίτητο να αποφύγει τη παγίδευση σε τοπικά βέλτιστα. Όταν η θερμοκρασία μειώνεται, το κόστος αρχίζει να μειώνεται. Η απόδοση του Timberwolf συγκρίθηκε με ένα πρόγραμμα που αναπτύχθηκε εμπορικά για τη τοποθέτηση και βασίζεται εν μέρει στην αλγόριθμο min-cut. Με τον TmberWolf επιτεύχθηκε 17-37% μικρότερο μήκος

σύρματος για βιομηχανικά κυκλώματα που κυμαίνονται 469 έως 2500 κελιά. Το 2500-κελιών κύκλωμα απαιτεί 15 ώρες χρόνου της CPU σε ένα IBM 3081K. Σε σύγκριση με χειροκίνητη διάταξη για ένα 800-κελί κύκλωμα, με τον Timberwolf επιτεύχθηκε μια κατά 24% μείωση του μήκους του καλωδίου με 4 ώρες του χρόνου της CPU σε ένα IBM 3081K.

1.4 Πρόσφατες Βελτιώσεις στο Simulated Annealing

Πρόσφατα οι ερευνητές έχουν αρχίσει να αναλύουν την απόδοση του αλγορίθμου και να ελέγχουν την λειτουργία των παραμέτρων χρησιμοποιώντας στατιστικές μεθόδους. Έχει αναφερθεί ως και δεκαπλάσια επιτάχυνση σε σχέση με τις προηγούμενες εκδόσεις.

1.4.1 Αποτελέσματα των Πιθανοτικών Συναρτήσεων Αποδοχής

Οι Nahar, Sahni, και Shragowitz [1985] πειραματίστηκαν με τις 20 διαφορετικές πιθανοτικές συναρτήσεις αποδοχής και τα προγράμματα θερμοκρασίας που παρατίθενται εδώ. Στη λίστα, g_k είναι η συνάρτηση αποδοχής, C_i, και C_j είναι τα προηγούμενα και νέα κόστη, και το T_k είναι το $\kappa_{\sigma\tau o}$ βήμα στη θερμοκρασία.

(1) Metropolis	$g_1 = \exp\left[-(C_j - C_i)/T_1\right]$	
(2) Έξι θερμοκρασίες Metropolis	$g_k = \exp\left[-(C_j - C_i)/T_k\right]$	1 ≤ k ≤6
(3) Συνεχής	$g_1 = 1$	
(4) Βήμα Μονάδας	$g_1 = 1$ και $g_2 = 0.5$	
(5) Γραμμική	$g_1 = T_1 C_i$	
(6) Τετραγωνική	$g_1 = T_1 C_i^2$	
(7) Κυβική	$g_1 = T_1 C_i^3$	
(8) Εκθετική	$g_1 = \frac{\exp\left(\frac{C_i}{T_1}\right)-1}{e-1}$	
(9) Έξι θερμοκρασίες γραμμικές	$g_k = T_k C_i$	1 ≤ k ≤6
(10) Έξι θερμοκρασίες τετραγωνικές	$g_k = T_k C_i^2$	1 ≤ k ≤6
(11) Έξι θερμοκρασίες κυβικές	$g_k = T_k C_i^3$	1 ≤ k ≤6
(12) Έξι θερμοκρασίες εκθετικές	$g_k = \frac{\exp\left(\frac{C_i}{T_k}\right)-1}{e-1}$	1 ≤ k ≤6
(13) Γραμμική διαφορά	$g_1 = T_1/(C_j - C_i)$	

(14) Τετραγωνική διαφορά $g_1 = T_1/(C_j - C_i)^2$

(15) κυβική διαφορά $g_1 = T_1/(C_j - C_i)^3$

(16) Εκθετική διαφορά $g_1 = \frac{\exp(T_1/(C_j - C_i)) - 1}{e-1}$

(17) Έξι θερμοκρασίες γραμμικής διαφοράς $g_k = T_k/(C_j - C_i)$ $1 \leq k \leq 6$

(18) Έξι θερμοκρασίες τετραγωνικής διαφοράς $g_k = T_k/(C_j - C_i)^2$ $1 \leq k \leq 6$

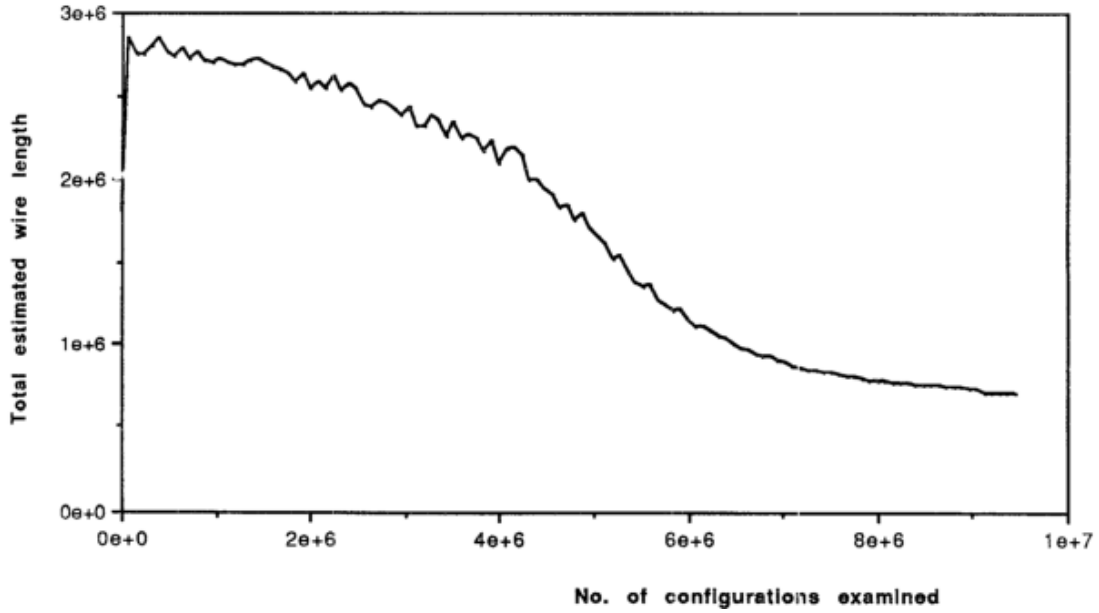
(19) Έξι θερμοκρασίες κυβικής διαφοράς $g_k = T_k/(C_j - C_i)^3$ $1 \leq k \leq 6$

(20) Έξι θερμοκρασίες εκθετικής διαφοράς $g_k = \frac{\exp(T_k/(C_j - C_i)) - 1}{e-1}$ $1 \leq k \leq 6$

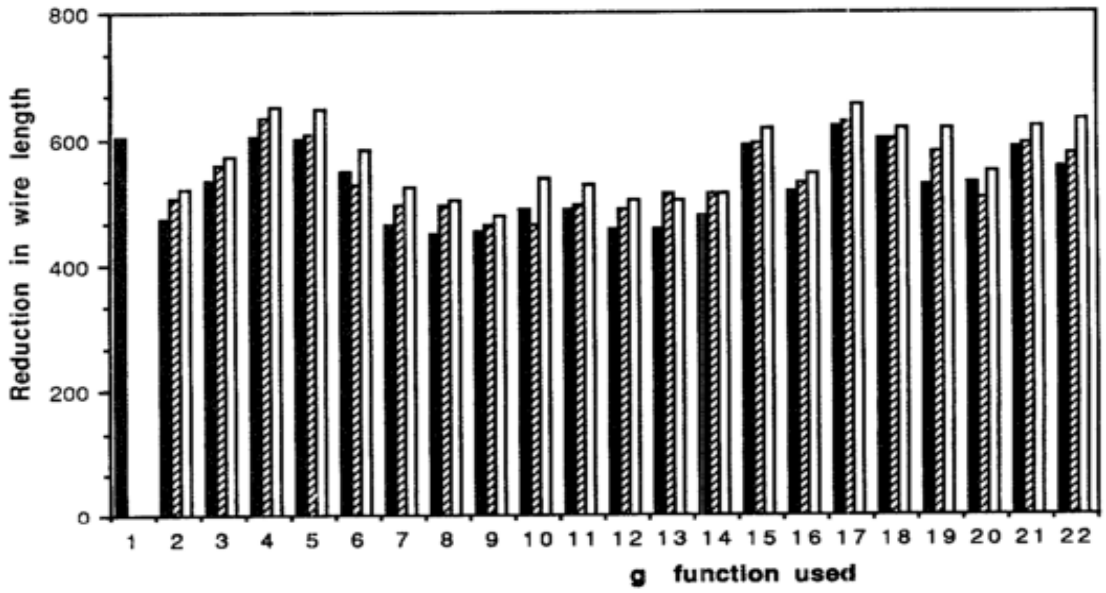
Για τη συνάρτηση του βήματος μονάδας και τις έξι συναρτήσεις της θερμοκρασίας, δόθηκε ίσος χρόνο υπολογισμού σε κάθε βήμα.

Αυτές οι συναρτήσεις δοκιμάστηκαν στο Net Optimal Linear Arrangement problem, που είναι κατά μια διάσταση ισοδύναμο του προβλήματος τοποθέτησης κελιών (cell placement). Σε όλες τις συναρτήσεις δόθηκε ίσος χρόνο υπολογισμού, καθώς γινόταν και σύγκριση στη μείωση του κόστους. Τα αποτελέσματα φαίνονται στο Σχήμα 10. Το σχήμα δείχνει επίσης μια σύγκριση με τον ευρετικό αλγόριθμο Goto [1977] τον Cohoon και τον Sahni [1983]. Η καλύτερη επίδοση παρουσιάστηκε από την άνοδο των έξι θερμοκρασιών, την σταθερά, και διάφορες κυβικές συναρτήσεις.

Εικόνα 9: Καμπύλη βελτιστοποίησης για TimberWolf 3.2



Εικόνα 10: Σύγκριση διαφόρων συναρτήσεων αποδοχής ■ 6 sec ▨ 9 sec □ 12 sec



1.4.2 Στατιστικός Έλεγχος των Annealing Parameters

Αν έχουμε μία μέθοδο για την εξαγωγή του χρονοδιαγράμματος των παραμέτρων ψύξης από μια στατιστική ανάλυση του ίδιου του προβλήματος, τότε το χρονοδιάγραμμα ψύξης, αντί να επανακαθορίζεται, μπορεί να προσαρμοστεί για το κάθε πρόβλημα που πρέπει να επιλυθεί, και το annealing μπορεί να προχωρήσει με ταχείς ρυθμούς. Τέτοιες προσεγγίσεις ονομάζονται adaptive simulated annealing algorithms. Ο Aarts et al. [1985, 1986] και ο van Laarhoven και ο Aarts [1987] χρησιμοποιούν τη θεωρία των αλυσίδων Markov για να αντλήσουν το χρονοδιάγραμμα ψύξης. Παρόμοιες εκφράσεις αναπτύχθηκαν από Huang et al. [1986].

Σημειογραφία

R = {r_1, r_2,,,,,,,,,,,,$r_{|R|}$ **}** είναι η διαμόρφωση του χώρου, το σύνολο όλων των πιθανών τοποθετήσεων, όπου το ***i*** , προσδιορίζει μονοσήμαντα μια διαμόρφωση, r_i, είναι η i_{th} διάταξη διανύσματος, δίνοντας τις συντεταγμένες όλων των ενοτήτων της **i-οστης** τοποθέτησης , e_i, είναι η i-οστη μονάδα στο διάστημα $[0, 1]^{|R|}$.

$I_R = \{i | r_i \epsilon R\} = \{1, 2, \ldots, i, \ldots, |R|\}$ είναι το σύνολο των ετικετών διαμόρφωσης, C:**R** →R είναι η συνάρτηση κόστους, η οποία αποδίδει ένα πραγματικό αριθμό **C(**r_i**)** για κάθε διαμόρφωση $i \epsilon I_R$ έτσι ώστε όσο χαμηλότερη η τιμή του C, τόσο καλύτερη είναι η αντίστοιχη διαμόρφωση. Ο αλγόριθμος μπορεί να μορφοποιηθεί ως μια ακολουθία αλυσίδας Markov, κάθε αλυσίδα αποτελείται από μια ακολουθία διαμορφώσεων για τις οποίες η πιθανότητα μετάβασης από τη διαμόρφωση ***i*** στη διαμόρφωση ***j*** δίνεται από τον τύπο

$$\Theta_{ij}(T)$$

$$= \begin{cases} A_{ij}(T)\,P_{ij} & \text{if } i \neq j \\ 1 - \sum\limits_{\substack{k=1 \\ k \neq i}}^{|\mathbf{R}|} A_{ik}(T)\,P_{ik}(T) & \text{if } i = j, \end{cases}$$

όπου το $\mathbf{P_{ij}}$, είναι η πιθανότητα διαταραχής, δηλαδή, η πιθανότητα παραγωγής μιας διαμόρφωσης **j** από μια διαμόρφωση **ί** (ανεξάρτητος του Τ) . $\boldsymbol{A_{ij}(T)}$ είναι η πιθανότητα αποδοχής, δηλαδή, η πιθανότητα αποδοχής της διαμόρφωσης ***j*** εάν το σύστημα είναι σε διαμόρφωση ***I*** και Τ είναι η θερμοκρασία. Η πιθανότητα διαταραχής επιλέγεται ως

$$P_{ij} = \begin{cases} \dfrac{1}{|\mathbf{R}_i|} & \text{if } j \in I_{\mathbf{R}_i} \\ 0 & \text{if } j \notin I_{\mathbf{R}_i}, \end{cases}$$

όπου $\boldsymbol{R_i}$, είναι η διαμόρφωση υποχώρου για τη διαμόρφωση ***i*** , το οποίο ορίζεται ως το διάστημα όλων των διαμορφώσεων που μπορεί να επιτευχθούν από τη διαμόρφωση ***ί*** με μια μόνο διαταραχή. Αυτή είναι μια ομοιόμορφη κατανομή πιθανοτήτων για όλες τις διαμορφώσεις στον υπόχωρο. Η πιθανότητα αποδοχής επιλέγεται ως

$$A_{ij}(T) = \begin{cases} \exp\left(\dfrac{-\Delta C_{ij}}{T}\right) & \text{if } \Delta C_{ij} > 0 \\ 1 & \text{if } \Delta C_{ij} \leq 0, \end{cases}$$

όπου $\boldsymbol{\Delta C_{ij} = C(r_j) - Cr_i}$. Αυτή η έκφραση είναι γνωστή ως το κριτήριο Metropolis (Metropolis Criterion). Από τη θεωρία της αλυσίδας Markov προκύπτει ότι υπάρχει ένα μοναδικό διάνυσμα ισορροπίας **q (T)** $\boldsymbol{\epsilon}$ $[\mathbf{0},\mathbf{1}]^{|R|}$ που ικανοποιείται για όλα τα

$$\text{for all } i \in I_{\mathbf{R}}\text{:} \quad \lim_{L \to \infty} \mathbf{e}_i^T \Theta(T) = \mathbf{q}^T(T).$$

Αν ξεκινήσουμε από οποιαδήποτε διαμόρφωση, ***i***, και εκτελέσουμε **L** διαταραχές , με $L \to \infty$, τότε η πιθανότητα να καταλήξουμε σε κατάσταση ***j*** δίνεται από το $\boldsymbol{q_j(T)}$ του διανύσματος ισορροπίας. Έτσι, το διάνυσμα ισορροπίας **q (T)** δίνει την κατανομή της πιθανότητας για την εμφάνιση κάθε κατάστασης σε ισορροπία. Για τις τιμές των $\mathbf{P_{ij}}$ και $\boldsymbol{A_{ij}(T)}$ που δόθηκαν παραπάνω,

$$q_j(T) = q_0(T)\exp\left(\frac{-\Delta C_{i_0 j}}{T}\right),$$

όπου i_0 είναι η ετικέτα της βέλτιστης διαμόρφωσης και το $q_0(T)$ είναι ένας παράγοντας κανονικοποίησης που δίνεται από τον τύπο

$$q_0(T) = \frac{1}{\sum_{k=1}^{|\mathbf{R}|} \exp\left(-\frac{\Delta C_{i_0 k}}{T}\right)}.$$

Επίσης έχουμε:

$$\begin{aligned}
&\lim_{\substack{T\to 0^+ \\ L\to\infty}} \left(\mathbf{e}_i \Theta(T)\right)_j \\
&\quad = \lim_{T\to 0^+} q_j(T) \\
&\quad = \begin{cases} |\mathbf{R}_0|^{-1} & \text{if } j \in I_{\mathbf{R}_0} \\ 0 & \text{if } j \notin I_{\mathbf{R}_0}, \end{cases}
\end{aligned}$$

όπου το R_0 είναι το σύνολο των βέλτιστων διαμορφώσεων, δηλαδή, $R_0 = \{r_i \epsilon R \,|C(r_i) = C_{r_{i0}}\}$. Έτσι, για αλυσίδες Markov άπειρου μήκους, το σύστημα θα πετύχει μια από τις βέλτιστες διαμορφώσεις με μία ομοιόμορφη κατανομή πιθανοτήτων, και η πιθανότητα της επίτευξης υποβέλτιστης (suboptimal) διαμόρφωσης είναι μηδέν.

Αρχική θερμοκρασία. Μια σταθερή αρχική θερμοκρασία T_1 δεν χρησιμοποιείται. Αντ 'αυτού, η αρχική θερμοκρασία έχει ρυθμιστεί έτσι ώστε να επιτευχθεί μία επιθυμητή αρχική πιθανότητα αποδοχής, x_0. Εάν το m_1 και το m_2 είναι ο αριθμός των μέχρι τώρα διαταραχών που οδηγούν σε μείωση του κόστους και αύξηση του κόστους, αντιστοίχως, και αν το αυξανόμενο κόστος διαταραχών m_2 είναι αποδεκτό σύμφωνα με το κριτήριο Metropolis, τότε ο συνολικός αριθμός των διαμορφώσεων που είναι αποδεκτός είναι m_1**+** m_2 ***exp (-ΔC / T)*** . Αυτό δίνει το x_0 ως

$$\chi_0 = \frac{m_1 + m_2 \exp\left(-\Delta C / T\right)}{m_1 + m_2}.$$

Η εξίσωση μπορεί να ξαναγραφεί για τον υπολογισμό της αρχικής θερμοκρασίας από την επιθυμητή τιμή του x_0:

$$T_1 = \overline{\Delta C^{(+)}} \left[\ln\left(\frac{m_2}{m_2 \chi_0 - (1-\chi_0) m_1}\right)\right]^{-1},$$

όπου $\overline{\Delta C^+}$ είναι η μέση τιμή όλων των αυξήσεων στο κόστος, αγνοώντας τις μειώσεις στο κόστος. Το αρχικό σύστημα παρακολουθείται κατά τη διάρκεια ενός

αριθμού διαταραχών πριν από την έναρξη της πραγματικής διαδικασίας βελτιστοποίησης. Ξεκινώντας με $T_1 = 0$, μετά από κάθε διαταραχή μια νέα τιμή για την T_1 υπολογίζεται από την ανωτέρω έκφραση.

Σύμφωνα με τον Huang et al. [1986], το σύστημα θεωρείται αρκετά ζεστό όταν το Τ >>σ , όπου σ είναι η τυπική απόκλιση της συνάρτησης κόστους. Εξ ου και η αρχική θερμοκρασία λαμβάνεται ως $T_1 = k\sigma$, όπου ***k=-3/ln(P)***. Αυτό επιτρέπει στην αρχική θερμοκρασία ***T*** να είναι αρκετά υψηλή ώστε να δέχεται με πιθανότητα ***P*** μια διαμόρφωση με κόστος ***3σ*** χειρότερο από το τρέχον. Μία τυπική τιμή του ***k*** είναι 20 για ***P = 0,9***. Πρώτον, ο χώρος διαμόρφωσης έχει διερευνηθεί για τον προσδιορισμό της τυπικής απόκλισης της συνάρτησης κόστους. Η αρχική θερμοκρασία υπολογίζεται μετά.

Πτώση θερμοκρασίας. Οι περισσότερες εφαρμογές χρησιμοποιούν προκαθορισμένες μειώσεις θερμοκρασίας, κάτι που δεν είναι βέλτιστο για όλες τις διαμορφώσεις ενός κυκλώματος. Ένα τέτοιο χρονοδιάγραμμα ψύξης οδηγεί σε αλυσίδες Markov μεταβλητού μήκους. Ο Aarts et al. [1986] συνιστά τη χρήση σταθερού μήκους για τις αλυσίδες Markov. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας την παρακάτω εξίσωση μείωσης θερμοκρασίας:

$$T_{i+1} = T_i \left(1 + \frac{\ln(1+\delta) T_i}{3\sigma_i}\right)^{-1},$$

όπου σ_i , είναι η τυπική απόκλιση της συνάρτησης κόστους έως την θερμοκρασία T_i, και το δ είναι ένας μικρός πραγματικός αριθμός που είναι μέτρο για το πόσο κοντά είναι τα διανύσματα ισορροπίας q_0 μεταξύ τους ύστερα από δυο διαδοχικές επαναλήψεις:

$$\frac{1}{1+\delta} \leq \frac{q_0(T_i)}{q_0(T_i)} \leq 1 + \delta.$$

Ο Huang et al. [1986] χρησιμοποίησε το μέσο κόστος σε σχέση με την log-temprature καμπύλη για την καθοδήγηση της μείωσης της θερμοκρασίας έτσι ώστε το κόστος να μειώνεται με ομοιόμορφο τρόπο. Ως εκ τούτου,

$$T_{i+1} = T_i \exp\left(\frac{T_i \Delta C}{\sigma^2}\right).$$

η εξίσωση αυτή έχει προέλθει από την εξίσωση της κλίσης της καμπύλης ανόπτησης για σ^2/T^2 . Για να διατηρήσουμε τη quasi-equilibrium, η μείωση του κόστους θα πρέπει να είναι μικρότερη από τη πρότυπη απόκλιση του κόστους. Για ΔC=-λσ λ≤1,

$$T_{i+1} = T_i \exp\left(-\frac{\lambda T_i}{\sigma}\right).$$

Τυπικά, λ = 0,7. Ο λόγος T_{i+1}/T_i / Τ, δεν επιτρέπεται να πέσει κάτω από ένα ορισμένο κάτω όριο (τυπικά 0.5) προκειμένου να αποτραπεί η δραστική μείωση της θερμοκρασίας που προκαλείται από την επίπεδη ανόπτηση της καμπύλης σε υψηλή θερμοκρασία.

Κριτήριο Διακοπής. Το κριτήριο διακοπής δίνεται από τον Aarts et al. [1986] ως

$$\left| \frac{\partial \overline{C}_s(T_i)}{\partial T} \frac{T_i}{\overline{C}(T_1)} \right| < \epsilon_s,$$

όπου ϵ_s, είναι ένα μικρός θετικός αριθμός που ονομάζεται η παράμετρος στοπ , και η $\overline{C}(T_1)$ είναι η μέση τιμή της συνάρτησης κόστουT_1ς. Η κατάσταση αυτή βασίζεται στην προέκταση του ομαλοποιημένου μέσου κόστους $\overline{C_s}T_i$ που λαμβάνεται κατά τη διάρκεια της διαδικασίας βελτιστοποίησης. Αυτός ο μέσος όρος υπολογίζεται επί μια σειρά αλυσίδων Markov, ώστε να μειωθούν οι διακυμάνσεις της $\overline{C}(T_1)$.

Η πολυπλοκότητα πραγματικού χρόνου και τα πειραματικά αποτελέσματα. Ο αλγόριθμος Aarts et al. [1986] έχει πολυπλοκότητα ***O(|R| In|R|)***, όπου **|R|** προέρχεται από το μήκος των αλυσίδων Markov, και ο όρος στο ***In|R|*** είναι ένα άνω όριο για τον αριθμό των βημάτων της θερμοκρασίας. Ο μηχανισμός διατάραξης μπορεί να επιλεγεί προσεκτικά, έτσι ώστε το μέγεθος της διαμόρφωσης των υποχώρων να είναι πολυωνυμική σε σχέση με τον αριθμό των μεταβλητών του προβλήματος. Κατά συνέπεια, ο annealing algorithm μπορεί πάντα να σχεδιαστεί να έχει χρονική πολυπλοκότητα πολυωνυμική σε σχέση με τον αριθμό των μεταβλητών.

Ο αλγόριθμος Huang et al. [1986] έχει δοκιμαστεί σε κυκλώματα μεγέθους 183 εώς 800 κελιών. Οδηγεί σε εξοικονόμηση του χρόνου CPU κατά 16-57% σε σύγκριση με τον Timberwolf για περίπου την ίδια ποιότητα τοποθέτησης. Ο χρόνος CPU που χρειάστηκε είναι της τάξεως των 9 ωρών σε έναν VAX 11/780 για ένα κύκλωμα 800 κελιών, ενώ το ίδιο κύκλωμα απαιτεί 11 ώρες με Timberwolf 3.2.

1.4.3 Βελτιωμένο annealing algorithm σε TimberWolfSC 4.2

Ο Sechen και Lee [1987] εφάρμοσαν ένα γρήγορο αλγόριθμο simulated annealing ως μέρος του TimberWolfSC έκδοσης 4.2, ο οποίος είναι 9-48 φορές πιο γρήγορος από ό, τι της έκδοση 3.2. Ως συνέπεια αυτού του αλγορίθμου, η τοποθέτηση έως 3000 κελιά μπορεί να γίνει σε ένα Micro VAX II workstation σε λιγότερο από 24 ώρες χρόνο CPU. Οι παράμετροι που χρησιμοποιούν έχουν ως εξής.

Κόστος λειτουργίας. Η τυπική συνάρτηση κόστους αποτελείται από την ημιπερίμετρο του μήκους του καλωδίου, με ρυθμιζόμενα βάρη για κατακόρυφα και οριζόντια neta και ποινές για επικάλυψη και έλεγχο μήκους γραμμών έχουν εφαρμοστεί. Η κωδικοποίηση, ωστόσο, είναι πολύ πιο αποδοτική. Για παράδειγμα, η κινήσεις που προκαλούν μια μεγάλη ποινή απορρίπτονται χωρίς να χάσουμε χρόνο CPU για εκτεταμένο υπολογισμό του μήκος του καλωδίου.

Πέναλτι επικάλυψης. Κάθε σειρά χωρίζεται σε μη επικαλυπτόμενα bins. Η ποινή επικάλυψης C_2 είναι ίση με το άθροισμα της απόλυτης διαφοράς μεταξύ του πλάτους

bin, ***w(b)***, και του συνολικού πλάτους του κελιού που τέμνει το bin $\boldsymbol{w_c(b)}$. Η ποινή επικάλυψης δίνεται από τη σχέση $\boldsymbol{C_2 = W_2 P_0}$, όπου η ποσότητα της επικάλυψης δίνεται από

$$P_O = \sum_{\text{bins}} | w_c(b) - w(b) | .$$

Η συνάρτηση αυτή μπορεί να αξιολογηθεί γρήγορα, επειδή ο αλγόριθμος δεν χρειάζεται να κάνει αναζήτηση μέσω όλων των κελιών προκειμένου να καθορίσει την επικάλυψη. Το $\boldsymbol{w_c(b)}$ είναι γνωστό για όλα τα bins. Όποτε ένα κελί μετακινείται, το $\boldsymbol{w_c(b)}$ ενημερώνεται για όλα τα κελιά που επηρεάζονται.

Η διαδικασία του simulated annealing εξαρτάται έντονα από το βάρος, $\boldsymbol{W_2}$, που δίνεται σε αυτή τη ποινή στο συνολικό κόστος λειτουργίας. Ως εκ τούτου, ένα αρνητικό σύστημα ανάδρασης έχει ενσωματωθεί για τον έλεγχο αυτής της παραμέτρου δυναμικά καθώς προχωρά το annealing:

$$W_2(i+1) = \max\left(0,\, W_2(i) + \frac{P_O - P_O^T}{\hat{L}_R}\right),$$

όπου $\boldsymbol{P_0}$ και $\boldsymbol{P_0^T}$ είναι η πραγματική και η τιμή στόχος της ποινής επικάλυψης, και $\boldsymbol{\hat{L}_R}$ είναι το επιθυμητό μήκος γραμμής. Αυτό αυξάνει την ποινή, εάν η επικάλυψη είναι μεγαλύτερη από την τιμή στόχος, διαφορετικά το μειώνει. Η ιδανική τιμή στόχος της επικάλυψης έχει εμπειρικά καθοριστεί:

$$P_O^T = \left[1.4 - 1.15\frac{i}{i_{\max}}\right]\hat{L}_R,$$

όπου $\boldsymbol{i}$ είναι η τρέχουσα επανάληψη, και $\boldsymbol{i_{max}}$ είναι ο αριθμός των επαναλήψεων (Τιμές θερμοκρασίας) που χρησιμοποιείται. Αυτό δίνει μια τιμή στόχος $\boldsymbol{1.4\ \hat{L}_R}$ σε υψηλές θερμοκρασίες, όταν $\boldsymbol{i \ll i_{max}}$. Καθώς η θερμοκρασία μειώνεται, ο έλεγχος γίνεται πιο σφιχτός και ο στόχος επικάλυψης μειώνεται μέχρι η τελική θερμοκρασία γίνει $\boldsymbol{0.25\hat{L}_R}$.

Ποινή στον Έλεγχο του Μήκους Γραμμής. Μια παρόμοια αρνητική ανατροφοδότηση στο δυναμικό έλεγχο έχει χρησιμοποιηθεί για τη συνάρτηση έλεγχου της ποινή στο μήκος γραμμής $\boldsymbol{C_3 = W_3 P_R}$, όπου το $\boldsymbol{P_R}$ δίνει τη διαφορά μεταξύ του πραγματικού και του επιθυμητού μήκους της σειράς. Βιομηχανικοί σχεδιαστές συστήνουν ότι η μέγιστη διακύμανση στα μήκη γραμμής από την επιθυμητή τιμή θα πρέπει να είναι μέσα στο 3%. Το πρόγραμμα προσπαθεί να επιτύχει αυτό το όριο, με τη συνεχή μεταβολή του βάρους $\boldsymbol{W_3}$. Η αρνητική ανατροφοδότηση της συνάρτησης ελέγχου είναι παρόμοια με εκείνη για την ποινή επικάλυψης:

$$W_3(i+1) = \max\left(0,\, W_3(i) + \frac{P_R - P_R^T}{P_R^T}\right),$$

όπου $\boldsymbol{P_R}$ και $\boldsymbol{P_R^T}$: είναι οι πραγματική τιμή και η τιμή στόχος της ποινής, και

$$P_R^T = \left[5 - 4\frac{i}{i_{\max}}\right]\zeta\hat{L}_R,$$

όπου ζ είναι η διαφορά του μέσου μήκους γραμμής. Εδώ έχουμε τις αρχικές και τελικές τιμές των $\boldsymbol{P_R^T}$

$$P_R^T(0) = 5\zeta\hat{L}_R, \quad \text{and} \quad P_R^T(i_{\max}) = \zeta\hat{L}_R.$$

Πρόωρη Απόρριψη Νέων Κινήσεων. Ενώ αξιολογούμε κινήσεις , η ποινή υπολογίζεται πριν το μήκος του καλωδίου. Αν μια κίνηση επιβαρύνεται με μεγάλη ποινή, είναι πιθανό η κίνηση αυτή να απορριφθεί. Ως εκ τούτου, δεν υπάρχει νόημα για τον υπολογισμό του μήκους καλωδίου για τέτοιες κινήσεις. Ο υπολογισμός της ποινής κάνει όσο ένα κλάσμα του χρόνου που απαιτείται για τον υπολογισμό του μήκους καλωδίου, επομένως πρόωρη απόρριψη τέτοιων κινήσεων μειώνει σημαντικά τον χρόνο υπολογισμού . Για πρόωρη απόρριψη, η αλλαγή στη ποινή ***ΔP*** υπολογίζεται:

$$\Delta P = \Delta C_2 + \Delta C_3 = \Delta C - \Delta C_1.$$

Η πιθανότητα αποδοχής ***exp (- ΔC / T)*** , είναι μικρότερη από ένα κατώτερο όριο **ϵ**, όταν

$$\Delta P > |\Delta C_{1.\min}| + T|\ln\epsilon|,$$

όπου $\boldsymbol{\Delta C_{1\,min}}$ είναι η μεγαλύτερη μείωση του μήκους του καλωδίου που αναμενόταν από την τρέχουσα επανάληψη. Αν η ποινή που υπολογίζεται ικανοποιεί αυτή την ανισότητα, η αξιολόγηση τερματίζεται. Θα ήταν επιθυμητό να μεγιστοποιηθεί ο αριθμός των απορρίψεων νωρίς για να εξοικονομήσουμε χρόνο CPU. Αυτό, ωστόσο, αυξάνει επίσης τον αριθμό των λανθασμένων κινήσεων πρόωρης απόρριψης - που λανθασμένα τερματίστηκαν, ενώ θα έπρεπε να γίνουν δεκτές. Για το σκοπό αυτό, μια καλή εκτίμηση της αναμενόμενης μείωσης σε μήκους καλωδίου $\boldsymbol{\Delta C_{1\,min}}$ απαιτείται. Αν η μεγαλύτερη τιμή του $\boldsymbol{\Delta C_{1\,min}}$ ήταν στην προηγούμενη επανάληψη, αυτή χρησιμοποιείται ως εκτίμηση, το σφάλμα είναι αρκετά μεγάλο, αφού ουσιαστικά το $\boldsymbol{\Delta C_1}$ κυμαίνεται από επανάληψη σε επανάληψη. Για

$$|\Delta C_{1\,\min}(i)| = |\overline{\Delta C_1(i-1)}| + 1.3\sigma(i-1),$$

η πρόωρη απόρριψη του σφάλματος είναι μικρότερη από 1%, όπου $\overline{\Delta C_1(\iota-1)}$ και ***σ(i - 1)*** είναι η μέση τιμή και η τυπική απόκλιση όλων των αρνητικών τιμών των ***ΔC*** πριν την επανάληψη ***i*** . Με αυτή την τιμή της $\Delta C_{1\,min}(\iota)$ και με $\epsilon \approx 1/3$, παίρνουμε την ανισότητα για το τεστ της έγκαιρης απόρριψης

$$\Delta P > |\overline{\Delta C_1(i-1)}| + 1.3\sigma(i-1) + T.$$

Move Generation. Η προηγούμενη μέθοδος της διατήρησης μιας σταθερής αναλογίας μεταξύ των μετατοπίσεων στις διασυνδέσεις έχει διακοπεί. Η ακόλουθη διαδικασία χρησιμοποιείται για την παραγωγή κινήσεων.

Ένα κελί επιλέγεται τυχαία, και μία τυχαία θέση έχει επιλεγεί ως προορισμός. Αν ο προορισμός είναι κενός, γίνεται η τοποθέτηση, αλλιώς εκτελείται ανταλλαγή. Μια νέα συνάρτηση range-limiting έχει χρησιμοποιηθεί, η οποία περιορίζει τη κίνηση ενός κελιού στη γειτονιά του. Αυτό προκάλεσε μια δραματική βελτίωση στο ποσοστό αποδοχής κινήσεων, εξοικονομώντας έτσι χρόνο που σπαταλιόταν στην αξιολόγηση κινήσεων που στο τέλος θα έπρεπε να απορριφθούν.

Προφίλ θερμοκρασίας. Το προφίλ της θερμοκρασίας είναι το βασικό χαρακτηριστικό αυτού του αλγορίθμου. Η δραματική βελτίωση στην αποδοχή του ποσοστού των νέων κινήσεων λόγω της συνάρτησης που οδηγεί σε βελτιωμένες κινήσεις, αυτή καθιστά περιττό να ξεκινήσουμε τον αλγόριθμο με πολύ υψηλή θερμοκρασία. Το προφίλ που χρησιμοποιείται θερμοκρασία είναι

$$T_1 = 500$$

$T_{i+1} = 0.98T_i$ ***1<i<120***

(Συγκρίνετε με Timberwolf 3.2, όπου T_1 ***= 4.000.000***) Έτσι, ο ίδιος αριθμός βημάτων για τη θερμοκρασίας συμπυκνώθηκε σε ένα μικρότερο εύρος. Η τελική θερμοκρασία είναι αμετάβλητη.

Αποδοχή Ρυθμού Ελέγχου . Λόγω της ευρείας ποικιλίας των κυκλωμάτων που πρόκειται να τοποθετηθούν, ένα πρόγραμμα με μια σταθερή θερμοκρασία δεν παράγει πάντα κατάλληλη τιμή του ποσοστού αποδοχής των νέων συνθέσεων. Παρατηρήθηκε ότι ο ιδανικός ρυθμός αποδοχής ήταν 50% στην αρχή (***i = 0***) και μειώθηκε στο μηδέν σε χαμηλές θερμοκρασίες ($i = i_{max}$). Για να επιτευχθεί αυτός ο ρυθμός ποσοστού αποδοχής, αρνητικός έλεγχος ανάδρασης έχει παρασχεθεί. Το ιδανικό ποσοστό αποδοχής του προφίλ δίνεται από

$$\rho_i^T = 50\left(1 - \frac{i}{i_{\max}}\right).$$

Αυτό το προφίλ επιτυγχάνεται με την κλιμάκωση της αλλαγή στο κόστος, ***ΔC***:

$$\Delta C' = s\,\Delta C,$$

Όπου

$$s_{i+1} = s_i\left(1 + \frac{\rho_i - \rho_i^T}{40}\right),$$

όπου $\boldsymbol{\rho_i}$ και $\boldsymbol{\rho_i^T}$ είναι η πραγματική τιμή και η τιμή στόχος του ποσοστού αποδοχής. Αυτό αλλάζει το s κατά 2.5% s για 1% απόκλιση του $\boldsymbol{\rho_i}$ και $\boldsymbol{\rho_i^T}$.

Ο αλγόριθμος δοκιμάστηκε σε έξι βιομηχανικές κυκλωμάτων και βρέθηκε να είναι 9-48 φορές πιο γρήγορος από τον Timberwolf 3.2, με μια ελαφρώς καλύτερη τοποθέτηση. Ελέγχθηκε επίσης και από τα MCNC κριτήρια αξιολόγησης, καθώς και το μήκος καλωδίου που ελήφθη ήταν 10-20% καλύτερο από άλλους αλγορίθμους. Ο χρόνος που απαιτείται για την επίτευξη αυτής της βελτίωσης, ωστόσο, δεν δίδεται.

Κάποια άλλα σημαντικά αναφορές για τοποθέτηση κελιών από simulated annealing είναι Bannerjee και Jones [1986], GIDAS [1985], Greene και Supowit [1984], οι Grover [1987], Hajek [1988], Lam και Delosme [1988], Lundy και Mees [1984], Mallela και Grover [1988], Romeo και Sangiovanni-Vincentelli [1985], Romeo et al. [1984], and White [1984].

2. Τοποθέτηση Force-Directed

Οι Force-Directed αλγόριθμοι τοποθέτησης είναι πλούσιοι σε ποικιλία και διαφέρουν σε μεγάλο βαθμό στις λεπτομέρειες εφαρμογής [Hanan και Kurtzberg [1972a]. Ο κοινός παρονομαστής σε αυτούς τους αλγορίθμους είναι η μέθοδος υπολογισμού της θέση όπου ένα module πρέπει να τοποθετηθεί, προκειμένου να επιτευχθεί ιδανική τοποθέτηση . Η μέθοδος αυτή έχει ως ακολούθως.

Εξετάστε κάθε δεδομένη αρχική τοποθέτηση. Ας υποθέσουμε ότι τα modules που συνδέονται με τα neta εξασκούν μια ελκτική δύναμη το ένα πάνω στο άλλο (Σχήμα 11). Το μέγεθος της δύναμης που δημιουργείται μεταξύ οποιωνδήποτε δύο modules είναι ευθέως ανάλογη προς την απόσταση μεταξύ των μονάδων. όπως και στο νόμο του Hooke για τη δύναμη που ασκείται από τα τεντωμένα ελατήρια, η σταθερά αναλογίας είναι το άθροισμα των βαρών από όλα τα neta που τα συνδέουν άμεσα. Εάν τα modules σε ένα τέτοιο σύστημα επιτρεπόταν να κυκλοφορούν ελεύθερα, θα κινούνταν προς την κατεύθυνση της δύναμης μέχρι το σύστημα να επιτύχει ισορροπία σε κατάσταση ελάχιστης ενέργειας, δηλαδή, με τα ελατήρια σε ελάχιστη τάση (η οποία είναι ισοδύναμη με ελάχιστο μήκος σύρματος), και με μια μηδενική συνισταμένη δύναμη σε κάθε ενότητα. Ως εκ τούτου, οι μέθοδοι τοποθέτησης force-directed βασίζονται στην κίνηση των μονάδων στην κατεύθυνση της συνολικής δύναμης που ασκείται πάνω τους έως ότου αυτή η δύναμη είναι μηδέν.

Ας υποθέσουμε ένα module $\boldsymbol{M_i}$, είναι συνδεδεμένο με το module $\boldsymbol{M_j}$ από ένα net $\boldsymbol{n_{ij}}$ έχοντας βάρος $\boldsymbol{w_{ij}}$. Ας υποθέσουμε ότι $\boldsymbol{s_{ij}}$ αντιπροσωπεύει την απόσταση από το $\boldsymbol{M_i}$ στο $\boldsymbol{M_j}$. Τότε η καθαρή δύναμη στην ενότητα δίνεται από

$$\mathbf{F}_i = \sum_j w_{ij}\mathbf{s}_{ij}.$$

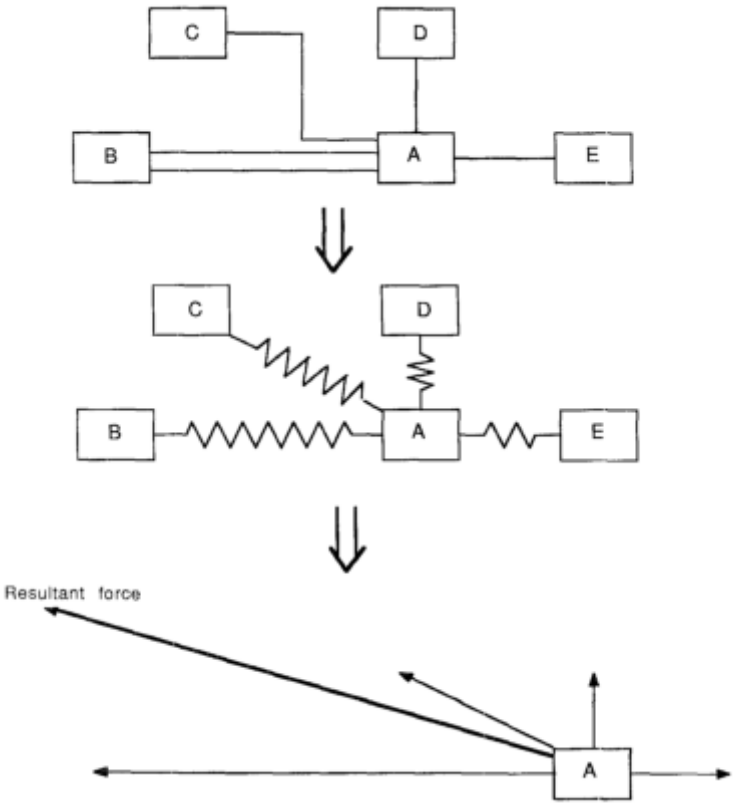
Εάν η χ και y συνιστώσες της δύναμης εξισώνονται στο μηδέν,

$$\sum_j w_{ij}(x_j - x_i) = 0,$$
$$\sum_j w_{ij}(y_j - y_i) = 0.$$

Έτσι, οι συντεταγμένες για το δυναμικό μηδενικό σημείο στόχου για το module $\boldsymbol{M_i}$, δίνεται από

$$\{x_i\}_{F=0} = \frac{\sum_j w_{ij}x_j}{\sum_j w_{ij}} = \bar{x}_i,$$
$$\{y_i\}_{F=0} = \frac{\sum_j w_{ij}y_j}{\sum_j w_{ij}} = \bar{y}_i.$$

Εικόνα 11: Η τοποθέτηση force-directed



Αυτές οι εξισώσεις επιδεικνύουν την ομοιότητα με τη βάση των εξισώσεων, δηλαδή, εάν τα τμήματα συνδέονται με το Mi, υποτίθεται ότι είναι μάζες που έχουν βάρος wij, τότε η θέση-στόχος μηδενικής δύναμης του Mi, είναι το κέντρο της βαρύτητας αυτών των τμημάτων.

2.1 Τεχνικές Τοποθέτησης Force-Directed

Οι πρώτες υλοποιήσεις του αλγόριθμου τοποθέτησης force-directed έγιναν στη δεκαετία του 1960 [Fisk et al. 1967]. Υπάρχουν πολλές παραλλαγές σε σχέση με σήμερα. Μερικές παραλλαγές είναι δομικοί αλγορίθμοι, ενώ κάποιοι βασίζονται σε iterative improvement.

Στις δομικές μεθόδους δεν υπάρχει καμία αρχική τοποθέτηση, οι συντεταγμένες τις κάθε μονάδας αντιμετωπίζονται ως μεταβλητές, και η καθαρή δύναμη που ασκείται σε κάθε μονάδα απο όλες τις άλλες μονάδες εξισώνεται με το μηδέν. Με την ταυτόχρονη επίλυση αυτών των εξισώσεων, παίρνουμε τις συντεταγμένες όλων των μονάδων. Σε μία τέτοια υλοποίηση, πρέπει να ληφθεί μέριμνα για να αποφευχθεί η τετριμμένη λύση xi = xj και yi = yj για όλα τα i, j, η οποία, λαμβάνοντας υπόψη το μοντέλο spring, ικανοποιεί προφανώς την συνθήκη μηδενικής δύναμης. Ένα άλλο πρόβλημα σε αυτή την προσέγγιση είναι ότι οι εξισώσεις μηδενικής δύναμης είναι μη γραμμικές, επειδή η δύναμη εξαρτάται από την απόσταση, με την Ευκλείδεια μετρική απόσταση να περιλαμβάνει τετραγωνική ρίζα, ενώ η απόσταση Μανχάταν

περιλαμβάνει απόλυτες τιμές. Ο Antreich at al. [1982] δίνει ένα παράδειγμα της μεθόδου επίλυσης τέτοιων εξισώσεων.

Στίς επαναληπτικές μεθόδους, μια αρχική λύση δημιουργείται είτε τυχαία ή από κάποια άλλη δομική μέθοδο. Τότε, μια μονάδα επιλέγεται σε κάθε χρόνο, υπολογίζεται το σημείο-στόχος μηδενικής δύναμης από τις παραπάνω εξισώσεις, και γίνεται προσπάθεια για να μετακινήσουμε τη μονάδα στο σημείο-στόχο ή να την εναλλάσσουμε με την μονάδα που προηγουμένως καταλαμβάνει το σημείο-στόχο. Αυτοί οι αλγόριθμοι ονομάζεται επίσης forced-directed relaxation ή forced-directed pairwise relaxation.

Εδώ, ένα πρόβλημα είναι, να αποφασίσουμε τη σειρά με την οποία θα επιλέξουμε τις μονάδες για τη μετάβαση τους στον τόπο προορισμού. Στις περισσότερες υλοποιήσεις, επιλέγεται η μονάδα με το ισχυρότερο διάνυσμα δύναμης. Σε άλλες υλοποιήσεις, οι μονάδες επιλέγονται τυχαία. Σε άλλες, οι μονάδες επιλέγονται με βάση την εκτίμηση της σύνδεσης τους.

Ένα άλλο πρόβλημα που υπάρχει είναι το πού να μετακινήσουμε την επιλεγμένη μονάδα εάν το διπλανό slot στη θέση-στόχο μηδενικής δύναμης είναι ήδη κατειλημμένο, που κατά πάσα πιθανότατα θα είναι. Μια λύση είναι να το μετακινήσουμε στη πλησιέστερη διαθέσιμη ελεύθερη θέση. Αλλά η πλησιέστερη ελεύθερη θέση μπορεί να είναι πολύ μακριά, σε ορισμένες περιπτώσεις. Αυτή είναι μια μέθοδος προσέγγισης και, στην καλύτερη περίπτωση, θα χρειαστούν περισσότερες επαναλήψεις για να επιτευχθεί μια καλή λύση.

Η δεύτερη λύση είναι να υπολογίσουμε τη θέση-στόχο μιας μονάδας όπως περιγράφεται παραπάνω, στη συνέχεια να αξιολογήσουμε την αλλαγή στο μήκος καλωδίου ή στο κόστος όταν η μονάδα εναλλάσσεται με τη αυτήν που βρίσκεται στη θέση στόχος. Εάν υπάρχει μείωση στο μήκος του καλωδίου, η ανταλλαγή γίνεται αποδεκτή, αλλιώς απορρίπτεται. Είναι αναγκαίο να αξιολογήσουμε το μήκος του καλωδίου επειδή είναι πιθανό σε μια προσπάθεια να γίνει εναλλαγή της επιλεγμένης μονάδας με μια μονάδα που βρίσκεται ήδη στο σημείο-στόχο της, να οδηγήσουμε την μονάδα που διώχνουμε πολύ μακριά και επομένως η κίνηση αυτή μπορεί να οδηγήσει σε απώλειες αντί για κέρδος.

Η τρίτη λύση είναι να εκτελέσουμε μια κίνηση κυμματισμού, δηλαδή, να επιλέξουμε τη μονάδα που προηγουμένως κατέλαβε το σημείο στόχος για την επόμενη κίνηση. Η διαδικασία αυτή συνεχίζεται έως ότου το σημείο στόχος μιας μονάδας βρεθεί σε ένα κενό slot. Στη συνέχεια, μια νέα μονάδα επιλέγεται.

Η τέταρτη λύση είναι να υπολογίσουμε το σημείο στόχος από κάθε μονάδα, στη συνέχεια, να αναζητήσουμε ζεύγη μονάδων, έτσι ώστε το σημείο στόχος μιας μονάδας να είναι πολύ κοντά στην τρέχουσα θέση της άλλης μονάδας. Εάν για αυτές τις ενότητες είναι δυνατή η εναλλαγή, τότε και οι δύο θα πάρουν την επιθυμητή θέση και θα έχουμε κοινό όφελος.

Η πέμπτη λύση χρησιμοποιεί επαναλαμβανόμενες δοκιμαστικές ανταλλαγές. Αν η ανταλλαγή μειώνει το κόστος, είναι αποδεκτό, αλλιώς θα απορριφθούν. Η συνάρτηση κόστους στην περίπτωση αυτή είναι το άθροισμα των δυνάμεων που επενεργούν επί

των ενοτήτων. Ένα παράδειγμα της χρήσης των δύο τύπων των συναρτήσεων δύναμης για ανταλλαγή ζευγών δίνεται στον Chyan και Breuer [1983].

Ο Hanan et al. [1976a, 1976b, 1978] συζητά και αναλύει επτά αλγόριθμους τοποθέτησης, συμπεριλαμβανομένων τριών με τεχνικές τοποθέτησης force-directed. Τα πειραματικά αποτελέσματα δίνονται στο Hanan [1976a], και οι αλγόριθμοι συζητούνται στο Hanan [1976b], Johannes et al. [1983], Quinn [1975], και στο Quinn and Breuer [1979] βρίσκονται εφαρμογές του αλγορίθμου force-directed.

2.2 Παράδειγμα

Θεωρήστε ένα κύκλωμα που αποτελείται από εννέα μονάδες, με την netlist να διαμορφώνεται ως εξής:

Net 1 = {1 3 4 8 9}

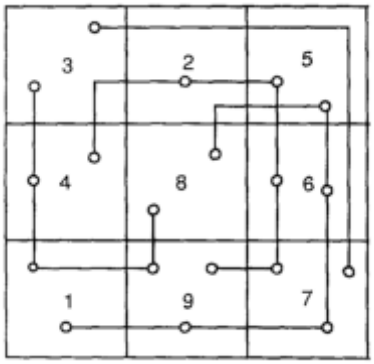
Net 2 = {1 5 6 7 8 9}

Net 3 = {2 4 5 6 7 9}

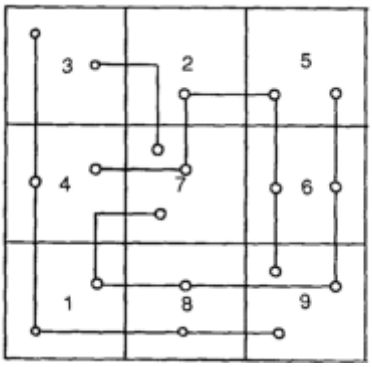
Net 4 = {3 7}

Το κατώτερο όριο για το μήκος του καλωδίου για το παράδειγμα αυτό είναι 15, υποθέτοντας ότι κάθε πήδημα από το ένα Net στο άλλο είναι 1 μονάδα (π.χ., το Net 1 πρέπει να είναι τουλάχιστον 4 μονάδες, προκειμένου να συνδεθούν σε αυτό πέντε τερματικοί σταθμοί). Για να δείξουμε πώς η force-directed τοποθέτηση δουλεύει , ξεκινάμε με μια τυχαία τοποθέτηση με ένα μήκος σύρματος 20, όπως φαίνεται στο Σχήμα 12α. Στον πίνακα 1 δίνεται η μήτρα σύνδεσης. Δύο επαναλήψεις φαίνεται λεπτομερώς στον Πίνακα 2. Στη πρώτη επανάληψη, το module 9 επιλέγεται σαν module τοποθέτησης, δεδομένου ότι έχει τη μεγαλύτερη συνδεσιμότητα, 14. Το σημείο στόχος είναι το (1,1,1), χρησιμοποιώντας το κέντρο του τύπου βαρύτητας με τις ενδείξεις στον Πίνακα 1 ως βάρη. Ως εκ τούτου το module 9 μετακινείται στη θέση (1, 1), αφήνοντας την αρχική του θέση (0, 1) κενή. Η τελευταία στήλη του Πίνακα 2 δίνει μια ενδιάμεση τοποθέτηση. Το module 8, που προηγουμένως βρισκόταν στο σημείο (1, 1), έχει επιλεγεί για την επόμενη κίνηση. Ο στόχος είναι το σημείο (0.9, 0.9), αλλά δεν μπορούμε να το τοποθετήσουμε στο (1,1) από τη στιγμή που ήδη το module 9 βρίσκεται εκεί. Ως εκ τούτου, αυτό τοποθετείται στο πλησιέστερο κενό slot του (0, 1). Τότε το module 7 επιλέγεται για τοποθέτηση, και η διαδικασία επαναλαμβάνεται. Το τελικό αποτέλεσμα φαίνεται στην Εικόνα 12b. Το αποτέλεσμα είναι μία βελτίωση στο μήκος σύρματος κατά 3 μονάδες.

Εικόνα 12: Παράδειγμα τοποθέτησης force-directed



(a) Τυχαία αρχική τοποθέτηση με καλώδιο μήκους 20



(b) Τελική τοποθέτηση μετά από δύο επαναλήψεις με καλώδιο μήκους 17

Πίνακας 1 Η μήτρα συνδεσιμότητας για το παράδειγμα της τοποθέτησης force-directed

Modules	1	2	3	4	5	6	7	8	9	Σ
1	0	0	1	1	1	1	1	2	2	9
2	0	0	0	1	1	1	1	0	1	5
3	1	0	0	1	0	0	1	1	1	5
4	1	1	1	0	1	1	1	1	2	9
5	1	1	0	1	0	2	2	1	2	10
6	1	1	0	1	2	0	2	1	2	10
7	1	1	1	1	2	2	0	1	2	11
8	2	0	1	1	1	1	1	0	2	9
9	2	1	1	2	2	2	2	2	0	14

Πίνακας 2: Οι πρώτες δύο επαναλήψεις για το παράδειγμα της τοποθέτησης force-directed

Iteration	Selected Module	Target Point	Case	Placed at	Result
1	9 (Seed)	(1.1, 1)	Occupied	(1, 1)	3 2 5 4 9 6 1 - 7
	8	(0.9, 0.9)	Locked	(1, 0)	3 2 5 4 9 6 1 8 7
	7 (Seed)	Locked	(1.1, 1.2)	Abort	3 2 5 4 9 6 1 8 7
	6 (Seed)	Locked	(1.2, 0.9)	Abort	3 2 5 4 9 6 1 8 7
2	9 (Seed)	(1.1, 0.9)	Same	Not moved	3 2 5 4 9 6 1 8 7
	7 (Seed)	(1.1, 1.2)	Occupied	(1, 1)	3 2 5 4 7 6 1 8 -
	9	(0.9, 1)	Locked	(2, 0)	3 2 5 4 7 6 1 8 9
	6 (Seed)	Locked	(1.2, 0.9)	Abort	3 2 5 4 7 6 1 8 9
	5 (Seed)	Locked	(1.2, 0.7)	Abort	3 2 5 4 7 6 1 8 9

2.3 Goto's Αλγόριθμος Τοποθέτησης

Ο αλγόριθμος Goto πρότεινε κατά κάποιο τρόπο κάτι μοναδικό για τους αλγόριθμους τοποθέτησης force-directed [Goto 1981, Goto and Matsuda 1986]. Αυτός ο αλγόριθμος αποτελείται από ένα αρχικό τμήμα τοποθέτησης και από ένα τμήμα επαναληπτικής βελτίωσης. Το τμήμα της αρχικής τοποθέτησης επιλέγει ενότητες για τοποθέτηση με βάση την συνδεσιμότητα. Όταν επιλέγεται μια μονάδα, τοποθετείται στη θέση που αποδίδει το ελάχιστο δυνατό μήκος καλωδίου και δεν μετακινείται κατά το υπόλοιπο της αρχικής φάσης τοποθέτησης.

Το τμήμα της επαναληπτική βελτίωσης χρησιμοποιεί μια γενικευμένη τεχνική force-directed relaxation στην οποία διερευνώνται οι εναλλαγές δύο ή περισσοτέρων μονάδων μέσα στην ε-γειτονιά (e-neighborhood) *της διαμέσου* μιας μονάδος. Διάμεσος μιας μονάδας ονομάζεται η θέση στην οποία το μήκος καλωδίου των δικτύων που συνδέονται με τη μονάδα είναι το ελάχιστο. Η ε-neighborhood της διάμεσου μιας μονάδας ορίζεται ως το σύνολο των ε-θέσεων για τη μονάδα, όπου το μήκος του καλωδίου που σχετίζεται με αυτή έχει τις μικρότερες e τιμές. Ο αλγόριθμος Goto μας δείχνει ότι το πρόβλημα εύρεσης της διαμέσου και της ε-neighborhood είναι διαχωρίσιμο σε x και y, και επομένως η x και y συντεταγμένες της διάμεσου μπορούν

να υπολογιστούν ανεξάρτητα το ένα από το άλλο χρησιμοποιώντας τον αλγόριθμο του Johnson και Mizoguchi [1978].

Η λ-neighborhood μιας δεδομένης διαμόρφωσης στο χώρο ορίζεται ως το σύνολο των συνθέσεων που μπορούν να ληφθούν από την συγκεκριμένη διαμόρφωση με κυκλική εναλλαγή και για όχι περισσότερες από λ μονάδες. Μια διαμόρφωση λέγεται ότι είναι λ-optimal (τοπικά βέλτιστη) αν είναι η καλύτερη δυνατή σε μια τέτοια γειτονιά. Η διαδικασία αντικατάστασης της τρέχουσας διαμόρφωσης με μια καλύτερη διαμόρφωση από την λ-neighborhood καλείται τοπική μεταμόρφωση (local transformation).

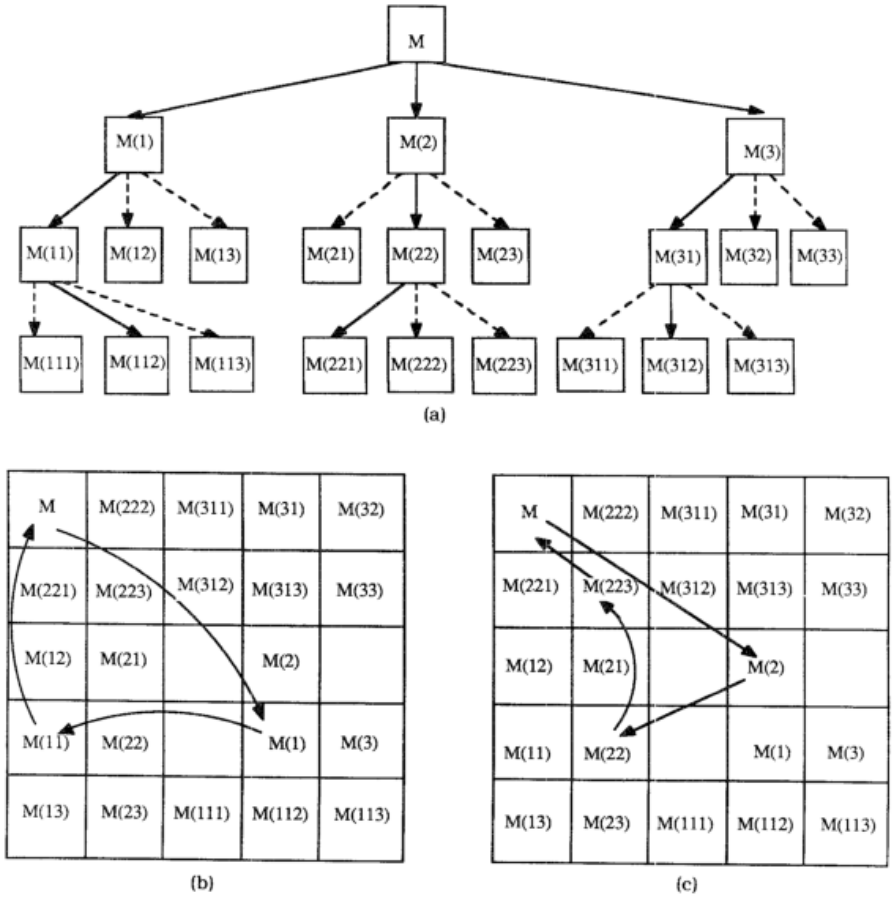
Ο πλήρης αλγόριθμος τοποθέτησης έχει ως ακολούθως. Μια πρώτη τοποθέτηση παράγεται. Εκτελείται γενικευμένος force-directed relaxation για να αποκτηθεί μια λ-optimal διαμόρφωση. Εάν ο δεδομένος χρόνος υπολογισμού δεν έχει εξαντληθεί, η διαδικασία επαναλαμβάνεται με άλλη αρχική τοποθέτηση. Το καλύτερο αποτέλεσμα όλων των δοκιμών είναι αυτό που γίνεται αποδεκτό. Ακολουθεί η ευρετική διαδικασία αναζήτησης που χρησιμοποιείται για την εύρεση της λ-optimal διαμόρφωσης.

Η διαδικασία αποτελείται από κύκλους ανταλλαγής μονάδων, που επαναλαμβάνονται έως ότου δεν υπάρχει περαιτέρω βελτίωση. Στην αρχή του κάθε κύκλου ανταλλαγής, μια μονάδα τοποθέτησης (M) επιλέγεται και εναλλάσσεται σε δοκιμαστική βάση με όλες τις ενότητες $M(i)$ στην ε-neighborhood (1 <i <ε). Εάν υπάρχει μια μείωση του μήκους καλωδίου, η εναλλαγή αποδίδει τη μέγιστη μείωση ως αποδεκτή, και ο κύκλος ανταλλαγής τερματίζεται. Εάν δεν υπάρχει μείωση του μήκους καλωδίου, μια τριπλή εναλλαγή δοκιμάζεται μεταξύ του module M που τοποθετείται , ένα module $M(i)$ από την ε-neighborhood και ενός module $M(ij)$ της ε-neighborhood του $M(i)$ (1 <i, j <ε). Αυτό έχει ως αποτέλεσμα ε^2 δοκιμές στις οποίες οι μονάδες εναλλάσσονται με κυκλική σειρά $M \rightarrow M(i) \rightarrow M(ij) \rightarrow M$. Εάν υπάρχει μια μείωση στο μήκος καλωδίου, τότε η ανταλλαγή που δίνει το ελάχιστο μήκος του καλωδίου γίνεται αποδεκτή, και ο κύκλος ανταλλαγής τερματίζεται. Διαφορετικά, για κάθε i, το $j = j_i$, που δίνει το ελάχιστο μήκος του καλωδίου θα επιλεγεί για περαιτέρω επεξεργασία. Το επόμενο βήμα είναι να προσπαθήσουμε ανταλλαγές τεσσάρων μεταξύ των M, $M(i)$, $M(ij_i)$ και της μονάδας $M(ij_ik)$ στην ε-neighborhood του $M(ij_i)$ (1 <i, k <ε). Αυτό πάλι οδηγεί σε ε^2 κόμβους της μορφής M $\rightarrow M(i) \rightarrow M(ij_i) \rightarrow M(ij_ik) \rightarrow$ M. Επιλέγουμε το k που οδηγεί στο ελάχιστο μήκος καλωδίου για περαιτέρω επεξεργασία. Η διαδικασία αυτή επαναλαμβάνεται έως ότου οι εναλλαγές των λ στοιχείων να έχουν εξεταστεί. Οι πιθανές ανταλλαγές εμφανίζονται ως ένα δέντρο στο σχήμα 13a. Οι εναλλαγές που οδηγούν στο ελάχιστο μήκος καλωδίου σε κάθε βήμα αντιπροσωπεύονται από τις συνεχείς γραμμές, ενώ εκείνες που αντιπροσωπεύονται από τις διακεκομμένες γραμμές εγκαταλείπονται. Υπάρχει μόνο μία συνεχής γραμμή από κάθε κόμβο εκτός της ρίζας M.

Η παράμετρος ε αντιπροσωπεύει το εύρος του δέντρου αναζήτησης, και το λ αντιπροσωπεύει το βάθος της. Καθώς τα ε και λ αυξάνονται, η λ-optimal διαμόρφωση γίνεται καλύτερη, αλλά υπάρχει επίσης μια μεγάλη αύξηση στον χρόνο υπολογισμού. Ο Goto αλγόριθμος παρατήρησε ότι ε = 4-5 και λ = 3-4 είναι ο καλύτερος συμβιβασμός μεταξύ ποιότητας τοποθέτησης και χρόνου υπολογισμού. Αυτά τα αποτελέσματα ελήφθησαν από πειράματα σε ένα κύκλωμα με 151 modules.

Για ικανοποιητική τοποθέτηση μεγαλύτερων κυκλωμάτων, μία υψηλότερη τιμή του ε και του λ μπορεί να είναι απαραίτητη.

Εικόνα 13: Σχέσεις force-directed (a) δένδρο αναζήτησης (b) ανταλλαγή λ = 3 (c) ανταλλαγή λ = 4



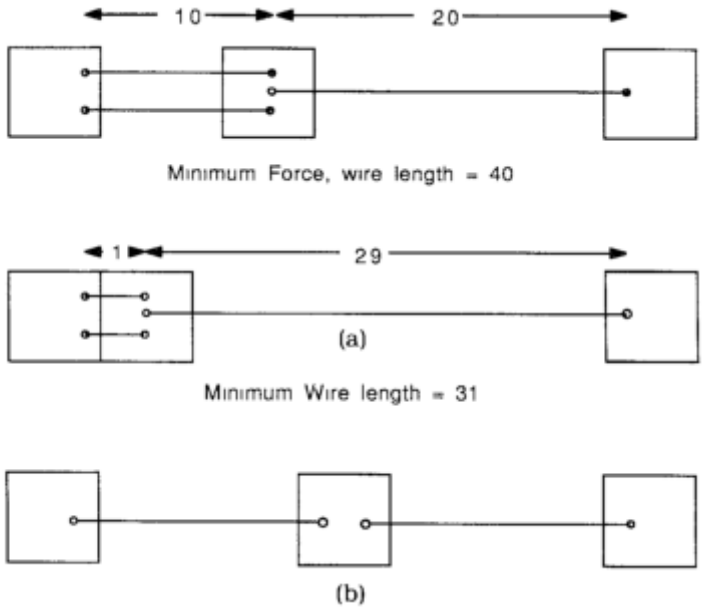
2.4 Ανάλυση

Μπορεί να αποδειχθεί ότι η ελάχιστη ενεργειακή κατάσταση του δυναμικού μοντέλου δεν δίνει πάντα το βέλτιστο μήκος σύρματος και αντιστρόφως. Ας εξετάσουμε το παράδειγμα στο Σχήμα 14α, όπου μια μονάδα συνδέεται με δύο neta από τα αριστερά και από ένα neto από τα δεξιά. Η τοποθεσία zero force θα είναι σε μια απόσταση 10 μονάδων από τα αριστερά και 20 μονάδων από τα δεξιά, αποδίδοντας ένα μήκος σύρματος 40. Για βέλτιστο μήκος καλωδίου, η μονάδα θα πρέπει να τοποθετηθεί άκρα αριστερά, δίδοντας ένα μήκος σύρματος μόνον 31. Ομοίως, θεωρείστε ένα module να συνδέεται με ένα neto από αριστερά και ένα neto από τα δεξιά (Σχήμα 14β). Αν και η μονάδα μπορεί να τοποθετηθεί οπουδήποτε και η τετμιμένη να μην επηρεάζει το μήκος του καλωδίου, η force-directed μέθοδος τοποθέτησης θα είναι άσκοπη περιοριζόμενη στη κεντρική τοποθεσία, εκτοπίζοντας ίσως κάποια άλλη ενότητα που πραγματικά θα έπρεπε να είναι στη θέση αυτή.

Λόγω της κληρονομικής φύσης του κέντρου της φόρμας βαρύτητας που χρησιμοποιείται, οι μέθοδοι force-directed τείνουν να τοποθετούν όλα τα modules

στο κέντρο του κυκλώματος. Το αποτέλεσμα είναι πάρα πολλοί δεσμοί και ακυρώσεις, με όλες τις ενότητες συνεχώς να εκτοπίζουν τα κεντρικά modules. Συνολικά, είναι μια σχετικά καλή μέθοδος τοποθέτησης μονάδων. Όταν τελειοποιηθεί σωστά και σε συνδυασμό με άλλες στρατηγικές που συζητήθηκαν παραπάνω, δίνει καλά αποτελέσματα. Αλλά είναι υποδεέστερη στην ποιότητα της λύσης με annealing simulation .

Εικόνα 14: Προβλήματα με την τοποθέτηση force-directed



3. Τοποθέτηση με Partitioning

Η τοποθέτηση με διαχωρισμό είναι μια σημαντική κατηγορία τοποθέτησης αλγόριθμων που βασίζονται σε επαναλαμβανόμενα τμήματα ενός δοθέντος κυκλώματος σε πυκνά συνδεδεμένα υποκυκλώματα έτσι ώστε ο αριθμός των δικτύων να μειωθούν από τη διαμέριση και να ελαχιστοποιηθούν. Επίσης, με κάθε διαμέριση του κυκλώματος, η διαθέσιμη περιοχή του τσιπ διαμοιράζεται εναλλάξ στην οριζόντια και την κάθετη κατεύθυνση (Σχήμα 15). Κάθε υποκύκλωμα θα ανατεθεί σε ένα διαμέρισμα της περιοχής του τσιπ. Εάν αυτή η διαδικασία συνεχιστεί μέχρις ότου κάθε υποκύκλωμα φτάσει να αποτελείται από μία μόνο μονάδα, τότε κάθε μονάδα θα έχει χαρτογραφηθεί σε μία μοναδική θέση του τσιπ. Οι περισσότερες τοποθετήσεις μέσω των partitioning αλγορίθμων, ή των Mini-cut αλγορίθμων, χρησιμοποιούν κάποια τροποποιημένη μορφή ευριτικών αλγορίθμων του Kernighan-Lin [1970] και του Fiduccia-Mattheyses [1982] heuristics για κατάτμηση, δείτε επίσης Schweikert και Kernighan [1972].

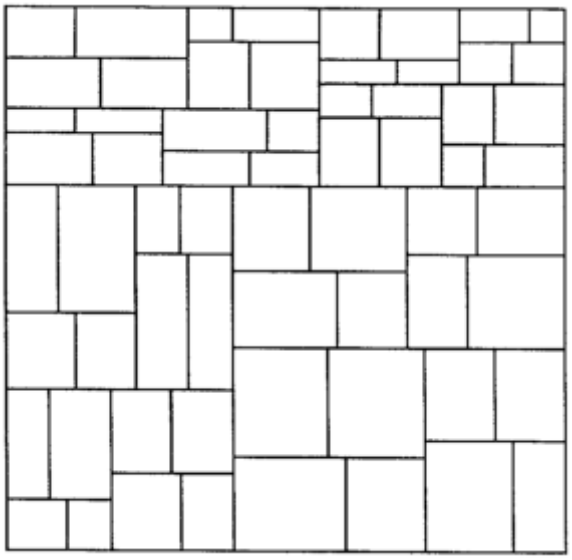
Ο αλγόριθμος του Kernighan-Lin για κατάτμηση έχει ως ακολούθως. Ξεκινήστε με ένα τυχαίο αρχικό χώρισμα που χωρίζει το σύνολο των μονάδων σε δύο ξένα σύνολα A και B. Αξιολογήστε το κόψιμο στα neta (ο αριθμός των δικτύων που συνδέουν ενότητες στην A και ενότητες στη B και ως εκ τούτου έχουν κοπεί από τη διαμέριση). Για όλα τα ζεύγη (a, b), $a \epsilon$A, $b \epsilon$B, βρείτε τη μείωση g στο net cut που λαμβάνεται από την εναλλαγή α και β (κινούμαι το a στο σετ B και το b στο σετ A). Το g ονομάζεται κέρδος της εναλλαγής. Αν g > 0, τότε η ανταλλαγή είναι ευεργετική. Επιλέξτε το ζεύγος μονάδας (a_1, b_1) με την υψηλότερο κέρδος g_1 . Αφαιρέστε το a_1 και το b_1 από τα A και B, και βρείτε το νέο ανώτατο όριο g_2 για μια ανταλλαγή κατά ζεύγη (a_2, b_2). Συνεχίστε αυτή τη διαδικασία μέχρις ότου τα A και B είναι άδεια. Βρείτε μια k τιμή τέτοια ώστε το συνολικό κέρδος να μεγιστοποιείται, και εναλλάσσονται

$$G = \sum_{i=1}^{k} g_i$$

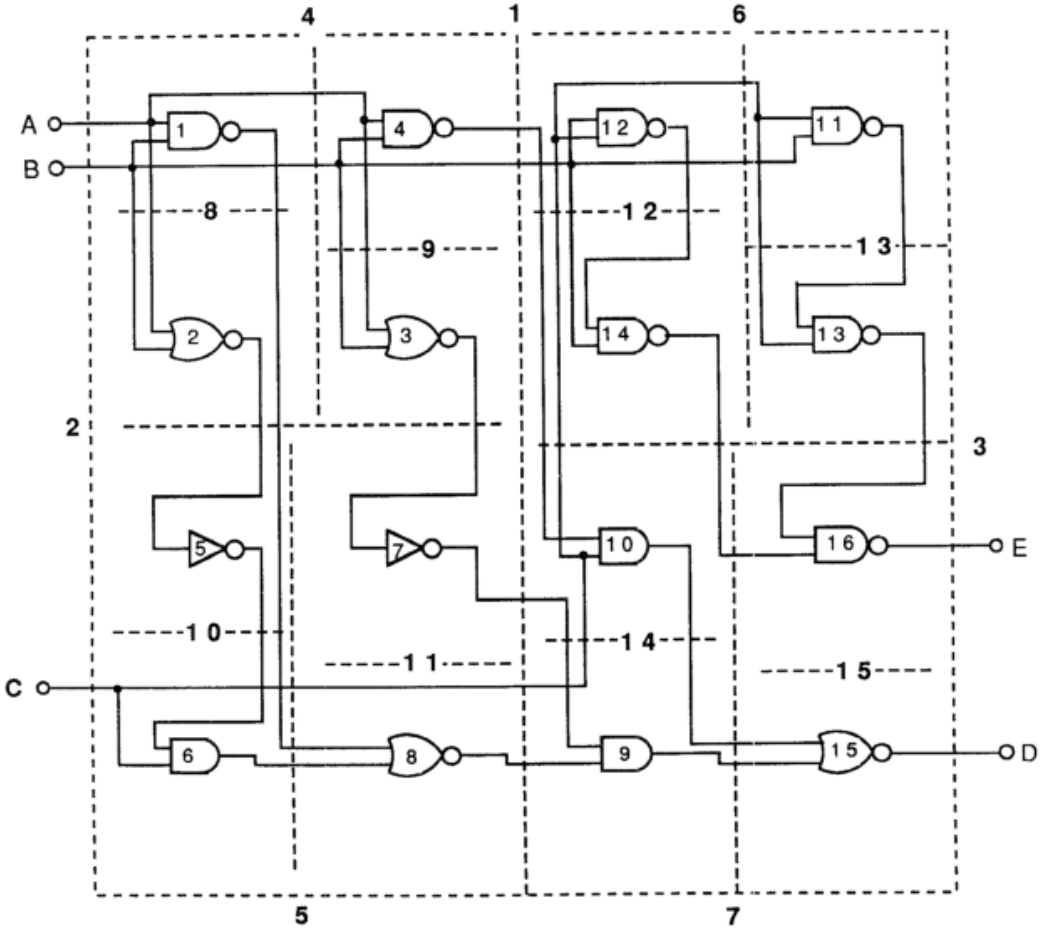
τα αντίστοιχα ζεύγη μονάδας $(a_1, b_1), (a_2, b_2) \ldots\ldots (a_\kappa, b_\kappa)$. Επαναλάβετε αυτή τη διαδικασία μέχρι G >0 και k>0.

Το Σχήμα 16 δείχνει ένα παράδειγμα τοποθέτησης με διαχωρισμό. Το σχήμα 4 δείχνει το κύκλωμα να τοποθετείται και τις επιθυμητές περιοχές για τα pads. Αυτό το κύκλωμα επανειλημμένα διαμερίζεται όπως φαίνεται στην Εικόνα 16. Σε κάθε στάδιο, ο αριθμός των δικτύων που τέμνονται από τη γραμμή-τομή ελαχιστοποιούνται, και τα υποκυκλώματα θα τοποθετηθούν σε οριζόντια ή κάθετα μέρη του τσιπ. Η προκύπτον τοποθέτηση (Σχήμα 4c) αποδίδει ένα συνολικό μήκος σύρματος 43 (για συνδέσεις αλυσίδων).

Εικόνα 15: Η περιοχή του τσιπ χωρισμένη σε κάθετη και οριζόντια διάταξη



Εικόνα 16: Διαμέριση Min-cut του κυκλώματος της εικόνας 4b.



3.1 Αλγόριθμος Breuer

Ο αλγόριθμος του Breuer [1977a, 1977b] είναι από τις πρώτες εφαρμογές με διαχωρισμού για την τεχνική της τοποθέτηση. Ελαχιστοποιείται ο αριθμός των δικτύων που κόβονται όταν το κύκλωμα κόβεται σε κάθε επανάληψη με δεδομένο ένα σύνολο γραμμών κοπής. Θεωρούμε μια σειρά από modules να συνδέονται με ένα σετ δικτύων. Έστω c είναι μια γραμμή που διασχίζει την επιφάνεια του τσιπ. Αν ένα ή περισσότερα στοιχεία συνδεδεμένα σε ένα neto s είναι στη μία πλευρά του c και ένα ή περισσότερα στοιχεία είναι από την άλλη πλευρά του c,τότε κατά τη δρομολόγηση του δικτύου ,τουλάχιστον μία σύνδεση πρέπει να διασχίσει τη γραμμή c. Η γραμμή κοπής c λέγεται ότι κόβει το neto s. Για μια δεδομένη τοποθέτηση, η τιμή του c, που συμβολίζεται με u(c) είναι ο συνολικός αριθμός των δικτύων που θα κοπούν από το c. Οι ακόλουθες συναρτήσεις αντικειμένων έχουν αναπτυχθεί για μια mini-cut τοποθέτηση:

(1) ***Ολικό net-cut*** .

Αυτή η συνάρτηση αντικειμένων θεωρεί το συνολικό αριθμό των δικτύων να κόβονται από όλες τις γραμμές-κοπής που διαμερίζουν το τσιπ.

$$N_c(\sigma) = \sum v(c),$$

όπου το άθροισμα είναι το σύνολο των κατακόρυφων και οριζόντιων γραμμών κοπής. Θεωρείστε ένα κανονικό σετ κοπής γραμμών ως τη συλλογή των γραμμών κοπής μεταξύ κάθε γραμμής και κάθε στήλης του κάθε slot. Στη συνέχεια, ελαχιστοποιούμε το συνολικό αριθμό των δικτύων που κόβονται χρησιμοποιώντας αυτό το σύνολο των γραμμών κοπής είναι ισοδύναμο με την ελαχιστοποίηση του semiperimeter (μισή-περίμετρος) μήκους καλωδίου. Για μια επίσημη απόδειξη, δείτε Breuer [1977a,1977b].

(2) Min-max cut value συνάρτηση αντικειμένων.

Στη τεχνολογία standard cell και gate array, το πλάτος του καναλιού, και επομένως η περιοχή του τσιπ, εξαρτάται από το μέγιστο αριθμός των δικτύων που δρομολογούνται μέσω ενός καναλιού σε οποιοδήποτε σημείο ή το μέγιστο net-cut για κάθε γραμμή κοπής διαμέσου του καναλιού. Η φόρμα αυτής της συνάρτηση αντικειμένων είναι

$$N_c(mM) = \sum_{\text{channels}} \max_{c \in C_i} v(c),$$

όπου C_i είναι ένα σύνολο γραμμών κοπής που ορίζονται σε ολόκληρη το i κανάλι. Σημειώστε ότι για αυτή τη συνάρτηση αντικειμένων , μόνο το net-cut στην πυκνή περιοχή συμφόρηση του καναλιού δρομολόγησης είναι σημαντική, και ο αλγόριθμος θα προσπαθήσει να ελαχιστοποιήσει αυτό το μέγιστο net-cut , ακόμη και εις βάρος της αύξησης του net-cut σε άλλες κενές περιοχές του καναλιού.

(3) Sequential cut-line συνάρτηση αντικειμένων.

Αν και οι παραπάνω συναρτήσεις αντικειμένων παρουσιάζουν καλύτερα το πρόβλημα τοποθέτησης , είναι υπολογιστικά δύσκολο να γίνει ελαχιστοποίηση. Μια τρίτη συνάρτηση αντικειμένων ως εκ τούτου παρουσιάζεται , η οποία είναι εύκολο να ελαχιστοποιηθεί, αλλά δεν δίνει βέλτιστη τοποθέτηση σε συνολικό επίπεδο. Όπως υποδηλώνει το όνομα της, το αντικείμενο της είναι να κάνει ένα κόψιμο και να ελαχιστοποιηθεί το net-cut, μετά να κόψει κάθε γκρουπ ξανά και να ελαχιστοποιήσει το net-cut με σεβασμό σε αυτές τις γραμμές-κοπής και με την επιφύλαξη των περιορισμών που ήδη επιβάλλονται από την προηγούμενη τομή, και ούτω καθεξής. Σημειώστε ότι, λόγω της διαδοχικής (άπληστη) φύσης αυτής της συνάρτησης, δεν εγγυάται ότι ο συνολικός αριθμός των δικτύων που θα κοπούν από όλες τις γραμμές κοπής θα ελαχιστοποιηθούν. Ως εκ τούτου, η ελαχιστοποίηση αυτής της συνάρτησης δεν είναι ισοδύναμη με την ελαχιστοποίηση του semiperimeter (μισή-περίμετρος) μήκους καλωδίου .

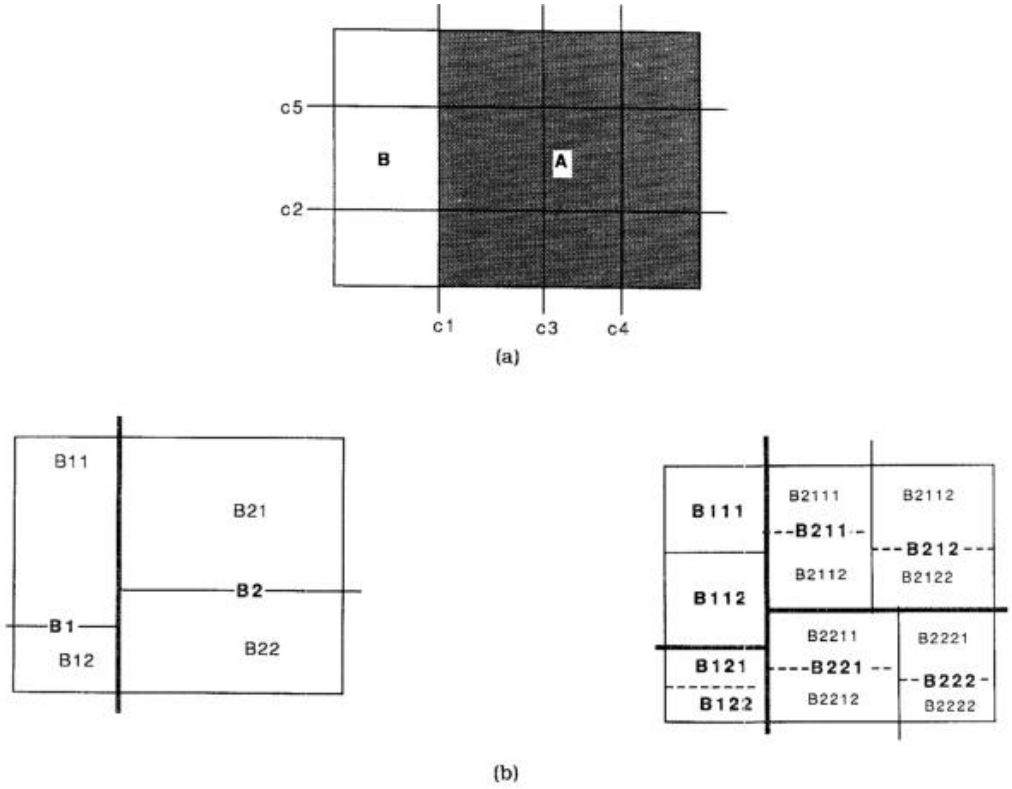
3.1.1 Αλγόριθμοι

Ο Breuer έχει διερευνήσει δύο βασικούς αλγορίθμους τοποθέτησης. Κάθε ένας από αυτούς τους αλγορίθμους απαιτεί μια δεδομένη ακολουθία γραμμών κοπής που διαχωρίζουν το τσιπ, έτσι ώστε κάθε τμήμα περιέχει μόνο ένα slot. Για να είμαστε συνεπής με το συμβολισμό του Breuer, στην ακόλουθη συζήτηση, τα υποτμήματα από το τσιπ που δημιουργήθηκαν από τη διαδικασία διαχωρισμού ονομάζονται blocks. Αυτά δεν πρέπει να συγχέεται με τα macro blocks .

Αλγόριθμος Cut Oriented Mini-Cut Placement. Ξεκινήστε με ολόκληρο το τσιπ και ένα δεδομένο σετ γραμμών κοπής. Αφήστε τη πρώτη cut-line να χωρίσει το τσιπ σε δύο blocks. Επίσης, χωρίστε το κύκλωμα σε δύο υποκυκλώματα τέτοια ώστε το net-cut να ελαχιστοποιείται. Τώρα χωρίστε όλα τα μπλοκ που τέμνονται από τη δεύτερη γραμμή τομής, και χωρίστε το κύκλωμα αντίστοιχα. Επαναλάβετε αυτή τη διαδικασία για όλες τις γραμμές κοπής. Η διαδικασία αυτή παρουσιάζεται στο Σχήμα 17a.

Ο αλγόριθμος αυτός αντιλαμβάνεται τη διαδοχική συνάρτηση αντικειμένων που περιγράφεται παραπάνω. Στην πράξη, ωστόσο, ο αλγόριθμος αυτός δεν δίνει πάντα καλά αποτελέσματα λόγω των δύο προβλημάτων που συνδέονται με αυτό. Εξετάστε το Σχήμα 17α. Ενώ κάνουμε την cut-line C_2 , πρέπει να χωρίσουμε τα μπλοκ Α και Β που δημιουργήθηκαν από το C_1 ταυτόχρονα. Πρώτον, αν υπάρχει ένας τρόπος για να τα χωρίσουμε διαδοχικά, ο χρόνος υπολογισμού θα αποθηκευτεί ως αποτέλεσμα της μείωσης του μεγέθους του προβλήματος. Εκτός αυτού, μια διαρροή μπορεί να προκύψει όταν προσπαθούμε να χωρίσουμε τα μπλοκ Α και Β χρησιμοποιώντας την ίδια γραμμή κοπής. Αν οι ενότητες του Α που τοποθετούνται πάνω από το C_2 απαιτούν μια μεγαλύτερη περιοχή από ό, τι τα αντίστοιχα στοιχεία στο Β, τότε είναι αδύνατον να διχοτομηθούν τα Α και Β με την ίδια γραμμή κοπής, και ένα μη βέλτιστη partitioning πρέπει να γίνει δεκτό. Για να αποφευχθεί αυτό το πρόβλημα, ένας άλλος αλγόριθμος παρουσιάζεται στον οποίο κάθε μπλοκ διαχωρίζεται χρησιμοποιώντας το καθένα μια ξεχωριστή γραμμή κοπής.

Εικόνα 17: Ο αλγόριθμος min-cut του Breuer.



Αλγόριθμος Block-Oriented Min-Cut Placement. Σε αυτόν τον αλγόριθμο, επιλέγουμε μια cut-line να διαχωρίσει το τσιπ σε δύο περιοχές. Στη συνέχεια, επιλέγουμε ένα ξεχωριστό cut-line για κάθε περιοχή και διαμερίζουμε τις περιοχές περαιτέρω. Η διαδικασία επαναλαμβάνεται έως ότου κάθε μπλοκ αποτελείται από μόνο ένα slot. Εδώ, διαφορετικές περιοχές μπορούν να έχουν διαφορετικές γραμμές κοπής, όπως φαίνεται στην Εικόνα 17b. Σημειώστε ότι πλέον δεν ελαχιστοποιούμε διαδοχικά τη συνάρτηση αντικειμένων, αφού δεν κάνουμε ομοιόμορφες περικοπές σε όλο το τσιπ.

Οι γραμμές κοπής για τη διαμέριση του τσιπ μπορεί να επιλεχθούν σε οποιαδήποτε αλληλουχία. Ο Breuer έχει δώσει τρεις αλληλουχίες (Σχήμα 18), οι οποίες είναι οι πλέον κατάλληλες για τρεις διαφορετικούς τύπους διατάξεων. Αυτά είναι ως εξής:

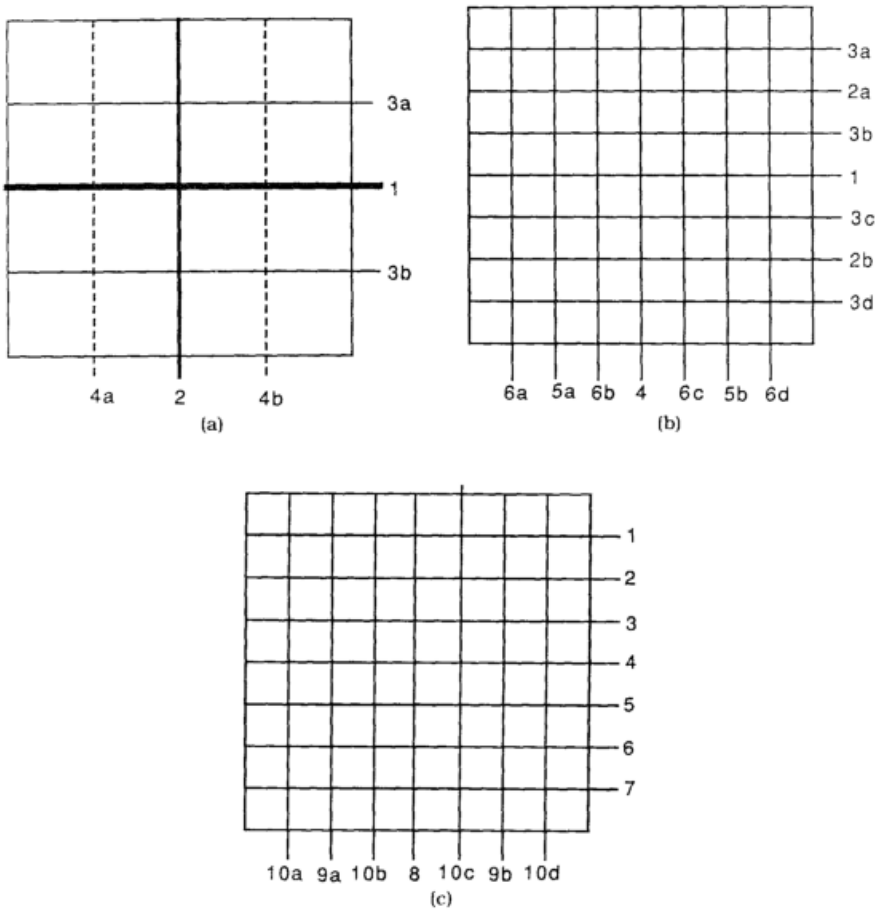
(1) Διαδικασία τοποθέτησης Quadrature.

Σε αυτόν τον αλγόριθμο η μέθοδος διαχωρισμού διεξάγεται πρώτα σε εύρος, με εναλλασσόμενα κάθετα και οριζόντια cuts . Αυτή η διαδικασία απεικονίζεται στο Σχήμα 18α. Με κάθε τομή, μια περιοχή υποδιαιρείται σε δύο ίσες υποπεριοχές . Αυτή η μέθοδος είναι κατάλληλη όταν υπάρχει υψηλή πυκνότητα δρομολόγησης στο κέντρο. Πρώτα γίνεται κοπή μέσα στο κέντρο και ελαχιστοποιώντας το net-cut, η συμφόρηση στο κέντρο μειώνεται. Αυτή είναι σήμερα η πιο δημοφιλής ακολουθία κοπής γραμμών για min-cut αλγόριθμο.

(2) Bisection Placement Procedure.

Στη διαδικασία αυτή, το τσιπ διχοτομείται (χωρίζεται σε δύο ίσες υποπεριοχές) επανειλημμένα με οριζόντιες γραμμές κοπής έως ότου κάθε υποπεριοχή να αποτελείται από μία μόνο σειρά. Αυτή η διαδικασία αναθέτει κάθε στοιχείο σε μια σειρά χωρίς να διορθώνει την θέσης του. Στη συνέχεια, κάθε γραμμή χωρίζεται σε δύο κατ 'επανάληψη μέχρι να προκύψει ότι κάθε υποπεριοχής περιέχει μόνο ένα slot, και έτσι όλα τα κινητά modules έχουν τοποθετηθεί (Σχήμα 18β). Αυτή είναι μια καλή μέθοδος για τοποθέτηση standard cell. Αυτό δεν σημαίνει, ωστόσο, ότι εγγυάται την ελαχιστοποίηση του μέγιστου net-cut ανά κανάλι.

Εικόνα 18: Η αλληλουχία χωρίσματος στον αλγόριθμο Breuer.



(3) Slice Bisection Procedure.

Μια άλλη στρατηγική τοποθέτησης είναι να διαχωρίσει ένα κατάλληλο αριθμό των μονάδων από τις υπόλοιπες του κυκλώματος και να τους τοποθετήσει σε μια σειρά (σε φέτες) με οριζόντια τομή γραμμής. Η διαδικασία αυτή επαναλαμβάνεται έως ότου κάθε μονάδα έχει ανατεθεί σε μια σειρά. Τότε οι μονάδες σε κάθε σειρά αποδίδεται σε στήλες με διχοτόμηση, χρησιμοποιώντας κάθετες γραμμές κοπής (Σχήμα 18c). Αυτή

η τεχνική είναι η πιο κατάλληλη όταν υπάρχει υψηλή πυκνότητα διασυνδέσεως στη περιφέρεια.

3.2 Dunlop Αλγόριθμος και Terminal Propagation

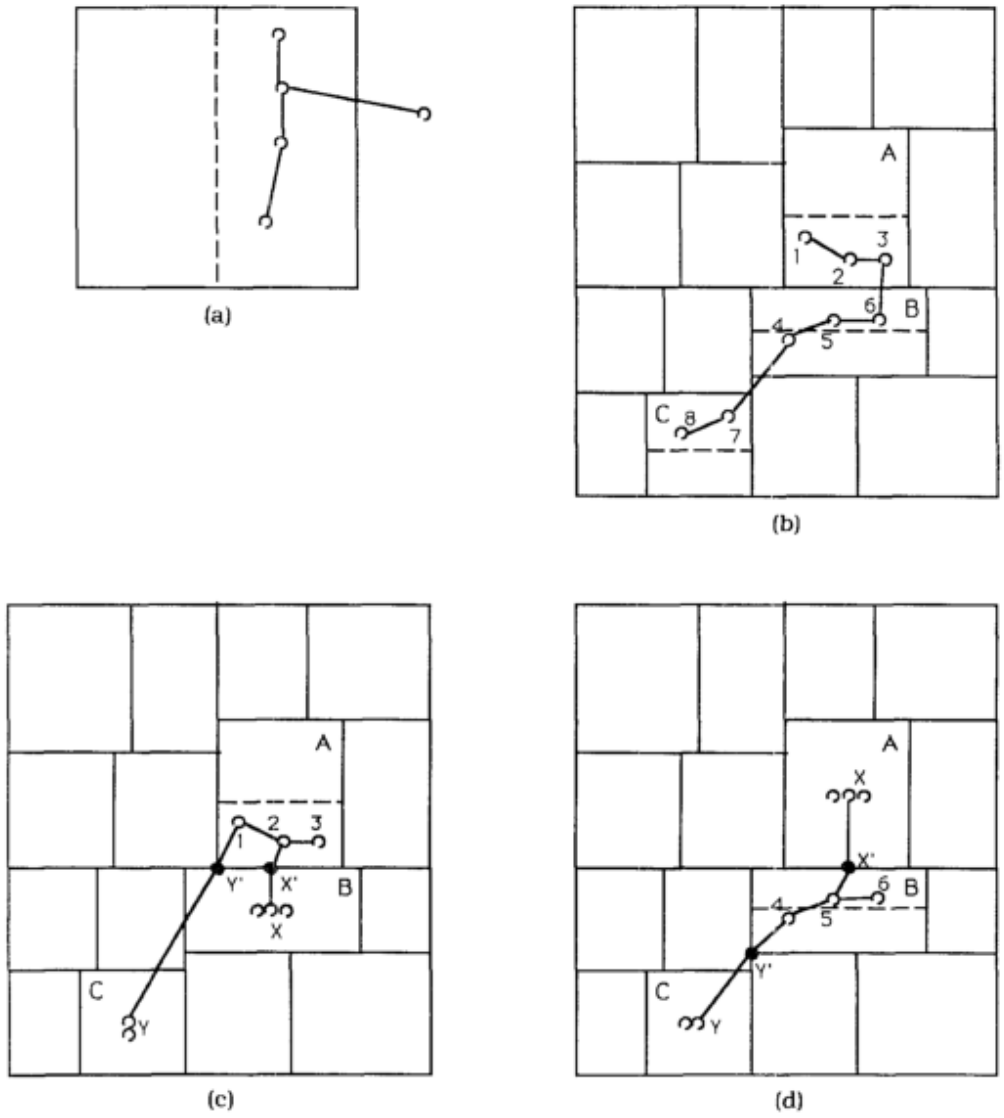
Όταν διαμερίζουμε ένα κύκλωμα ή ένα τμήμα ενός κυκλώματος σε δύο μέρη, δεν είναι αρκετό να εξετάσουμε μόνο τα εσωτερικά neta του κυκλώματος, τα οποία μπορεί να τέμνονται με τη γραμμή κοπής. Διότι τα neta συνδέονται με εξωτερικά τερματικά ή με άλλα modules σε άλλα διαμερίσματα (σε υψηλότερο επίπεδο) και θα πρέπει επίσης να εξεταστούν. Ο Dunlop και Kernighan [1985] το έκαναν αυτό με μια μέθοδο που ονομάζεται διάδοση τερματικού(terminal propagation). Το Σχήμα 19 απεικονίζει την ανάγκη για διάδοση τερματικού. Το Σχήμα 19α δείχνει την πρώτη διαίρεση ολόκληρου του κυκλώματος σε δύο τμήματα. Αν ένα module είναι συνδεδεμένο σε ένα εξωτερικό τερματικό στη δεξιά πλευρά του τσιπ, θα πρέπει κατά προτίμηση να ανατεθεί στην δεξιά πλευρά του τσιπ, και αντιστρόφως. Αν αυτός ο περιορισμός δεν λαμβανόταν υπόψη , τότε κάθε ήμισυ του κυκλώματος θα μπορούσε να είχε ανατεθεί σε οποιαδήποτε πλευρά του τσιπ. Το Σχήμα 19β δείχνει το αποτέλεσμα μετά από αρκετές επίπεδα διαμέρισης. Ένα συγκεκριμένο neto έχει κελιά που συνδέονται σε αυτό σε τμήματα A, B, και C, όπως φαίνεται. Όταν αυτά τα τμήματα διαχωριστούν περαιτέρω, θα ήταν προτιμότερο να τοποθετήθουν αυτά τα κελιά στο κάτω μισό του A, αλλά στο πάνω μισό του C. Η ανάθεση στο B δεν επηρεάζει το μήκος σύρματος. Ο Dunlop και Kernighan [1985] εφάρμοσαν τη διάδοση τερματικού ως εξής.

Θεωρούμε την κατάσταση όταν A διαχωρίζεται κάθετα και τα net connecting cells 1, 2, και 3 στο A συνδέονται επίσης με άλλα κελιά στα B και C. Τέτοια κελιά σε άλλες κατατμήσεις υποτίθεται ότι είναι στο κέντρο των περιοχών των διαμερισμάτων τους (σημεία X και Y στο σχήμα 19c) και αντικαθίστανται από dummy κελιά στα πλησιέστερα σημεία στα σύνορα του A (π.χ , στο X ', Y'). Τώρα, κατά τη διάρκεια της κατάτμησης, το net-cut θα ελαχιστοποιηθεί αν τα κελιά 1, 2, και 3 τοποθετηθούν στο κάτω μισό του A. Μια παρόμοια διαδικασία για το B δεν θα αποδώσει καμία προτίμηση (Σχήμα 19d), όπως προβλέπεται από τα παραπάνω.

Για να κάνουμε διάδοση τερματικού σταθμού, ο διαχωρισμός πρέπει να γίνει πρώτα . Δεν υπάρχει λόγος διαχωρισμού μιας ομάδας σε λεπτότερα και ακόμη λεπτότερα επίπεδα χωρίς να γίνει διαχωρισμός και των άλλων ομάδων, αφού σε αυτή την περίπτωση δεν θα υπάρχουν πληροφορίες διαθέσιμες σχετικά με το σε πιο γκρουπ μια μονάδα πρέπει κατά προτίμηση να ανατεθεί.

Ο αλγόριθμος είναι σε παραγωγική χρήση ως μέρος ενός αυτόματου συστήματος σχεδιασμού. Ο αλγόριθμος έχει δοκιμαστεί σε ένα τσιπ με 412 cells και 453 nets. Αποδίδει σε περιοχές εντός 10-20%. Ο χρόνου της CPU έχει αναφερθεί της τάξης της 1 ώρας σε έναν VAX 11/780. Ο χρόνος CPU μπορεί να βελτιωθεί σημαντικά χρησιμοποιώντας τον Fiduccia-Mattheyses [1982] γραμμικού χρόνου ευρετικό διαχωρισμό.

Εικόνα 19: Terminal Propagation



3.3 Quadrisection

Ο Suaris και Kedem [1987] έχουν προτείνει τη χρήση της quadrisection αντί της bipartitioning να διαιρέσει το τσιπ κάθετα και οριζόντια σε ένα ενιαίο στάδιο διαίρεσης (Σχήμα 20α), καταλήγοντας σε μια αληθινά δισδιάστατη διαδικασία τοποθέτησης, αντί να προσαρμόσουν μια μονοδιάστατη διαδικασία διαμέρισης για να λύσει το δισδιάστατο πρόβλημα τοποθέτησης. Ο quadrisection αλγόριθμος που χρησιμοποιείται είναι μια επέκταση του Kernighan-Lin [1970] και του Fiduccia-Mattheyses [1982].

Σε αντίθεση με το Kernighan-Lin αλγόριθμο που περιγράφεται παραπάνω, ένα module σε ένα τεταρτημόριο μπορεί να εναλλάσσεται με modules σε οποιαδήποτε από τα άλλα τρεις τεταρτημόρια. Αυτό δίνει 12 πίνακες κέρδος, έκαστο των οποίων

αντιστοιχεί σε ένα ζεύγος τεταρτημόρια. Σε κάθε βήμα, επιλέγονται αυτά τα ζεύγη για ανταλλαγή που δίνουν την υψηλότερο κέρδος.

Η συνάρτηση κόστους υπολογίζεται ως εξής. Αφήστε τα κελιά που συνδέονται με στο net n και τοποθετήθηκαν στο τεταρτημόριο K να σημανθούν ως $\boldsymbol{\alpha_K(n)}$. Τότε το διάνυσμα cell-distribution για το neto είναι

$$\boldsymbol{a(n) = \{a_1(n), a_2(n), a_3(n), a_4(n)\}}$$

Σε συνδυασμό με κάθε neto είναι διάνυσμα resident flag

$$\boldsymbol{\beta(n) = \{\beta_1(n), \beta_2(n), \beta_3(n), \beta_4(n)\}}$$

έτσι ώστε

$$\beta_K(n) = \begin{cases} 1 & \text{if } \alpha_K(n) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Έτσι, το K συστατικό του $\boldsymbol{\beta_K(n)}$ δείχνει κατά πόσο κάποιο κελί που συνδέεται με το neto ***n*** είναι στο τεταρτημόριο K. Η συνάρτηση κόστους ορίζεται ως

$$W = \sum_{n \in N} w_n\big(\beta(n)\big),$$

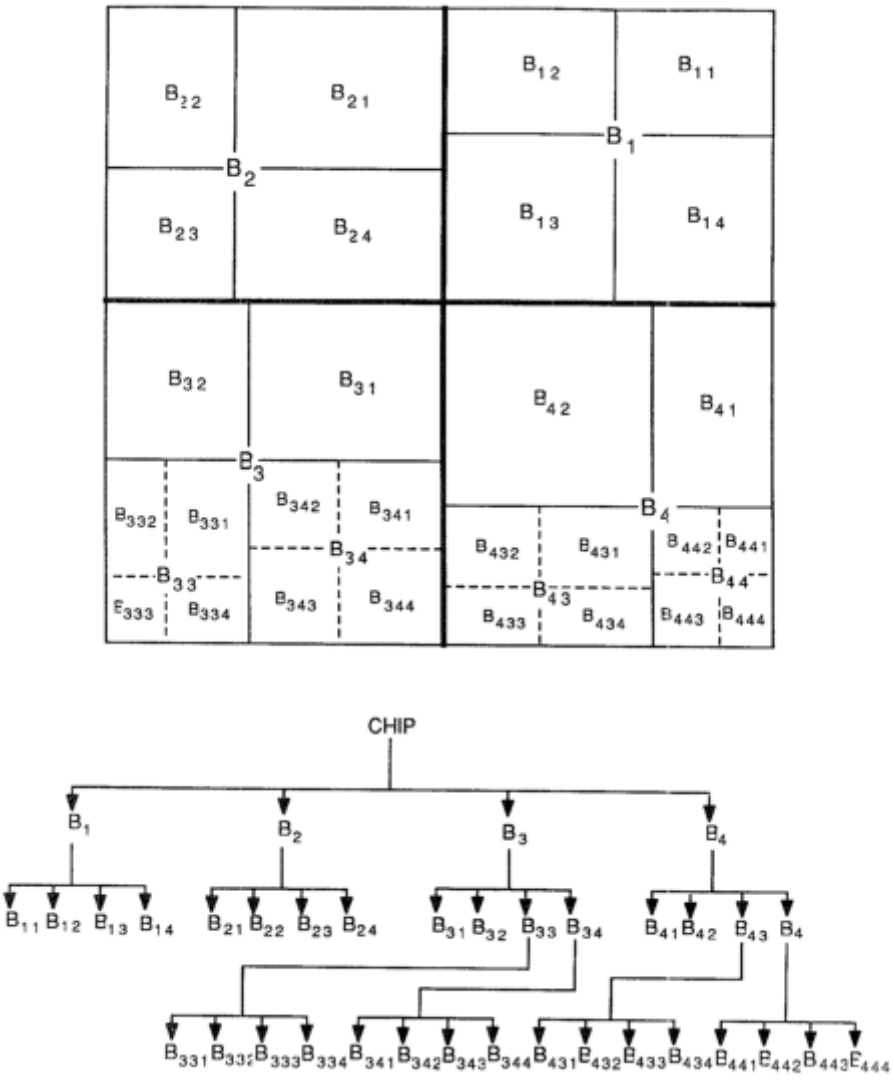
όπου $\boldsymbol{w_n(\beta(n))}$ είναι το κόστος του δικτύου ***n***. Εάν δύο ή περισσότερα συστατικά του ***β(n)*** είναι μη μηδενικά, τότε υπάρχουν κελιά που συνδέεται με το εν λόγω neto στα αντίστοιχα τεταρτημόρια, και το neto έχει κοπεί. Τα βάρη $\boldsymbol{w_h}$ και $\boldsymbol{w_v}$ συνδέονται με το οριζόντιο και κάθετο net-cut, αντίστοιχα. Οι σχετικές τιμές των εν λόγω βαρών δείχνουν την προτίμησή στην κατεύθυνση της καλωδίωσης. Σύμφωνα με τον Suaris και Kedem [1987], σε τεχνολογία διπλού και τριπλού μετάλλου, όπου σχεδόν το σύνολο του διαστήματος γύρω από τα κελιά μπορεί να χρησιμοποιηθεί για την καλωδίωση, θα προτιμούσαμε την κάθετη (πάνω από το κελί) καλωδίωση. Αυτό θα διατηρήσει χώρο για τον δίαυλο, ο οποίος διαφορετικά θα απαιτούνταν για ανοίγματα οριζόντιας καλωδίωσης. Έτσι, σε τέτοιες τεχνολογίες, το $\boldsymbol{w_v}$ συνήθως ορίζεται πολύ μικροτερο από ό, τι το $\boldsymbol{w_h}$.

Αν όλα τα modules που συνδέονται σε ένα neto είναι σε οριζοντίως γειτονικά τεταρτημόρια, τότε το κόστος $\boldsymbol{w_n\big(\beta(n)\big) = w_h}$. Παρομοίως, εάν είναι σε κατακόρυφα γειτονικά τεταρτημόρια, τότε $\boldsymbol{w_n\big(\beta(n)\big) = w_v}$. Εάν τα modules είναι διαγωνίως σε απέναντι τεταρτημόρια ή εάν έχουν διανεμηθεί σε παραπάνω από τρία τεταρτημόρια, τότε $\boldsymbol{w_n\big(\beta(n)\big) = w_h + w_v}$. Εάν τα modules συνδέονται σε ένα net n κατανέμονται πάνω και στα τέσσερα τεταρτημόρια, υπάρχουν δύο πιθανές διασύνδεσης μοτίβα, ένα με ένα οριζόντιο και δύο κατακόρυφα cuts και η άλλη με ένα κάθετο και δύο οριζόντια cuts. Εάν $\boldsymbol{w_v < w_h}$ όπως περιγράφηκε παραπάνω, θα επιλέξουμε το πρώτο μοτίβο και $\boldsymbol{w_n\big(\beta(n)\big) = w_h + 2w_v}$.

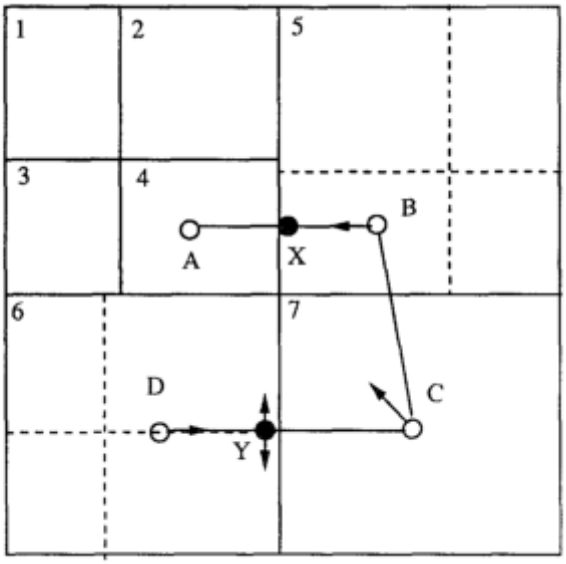
Εάν η συνάρτηση κόστους $\boldsymbol{w_n(\beta(n))}$ είναι τέτοια ώστε να μπορεί να υπολογιστεί από το ***β(n)*** σε γραμμικό χρόνο, μπορεί να αποδειχθεί ότι ο quadrisection αλγόριθμος τρέχει επίσης σε γραμμικό χρόνο. Το υπόλοιπο του αλγορίθμου διαμέρισης είναι ίδιο όπως στο Fiduccia και Mattheyses [1982] και Kernighan και Lin [1970].

Η μέθοδος διάδοσης τερματικού εισήχθη από τον Dunlop και Kernighan [1985] έχει επεκταθεί για το quadrisection όπως φαίνεται στο Σχήμα 20b. Το σχήμα δείχνει τα τμήματα 5 και 6 που πρόκειται να διαχωριστούν

Εικόνα 20a: Quadrisection



Εικόνα 20b: Terminal propagation σε Quadrisection



κατά μήκος των διακεκομμένων γραμμών και τα cells B και D σε αυτές τις περιοχές συνδέονται με τα cells A και C σε άλλες περιοχές. Σε αυτό το παράδειγμα, θα ήταν ωφέλιμο να θέσουμε το B στο κάτω αριστερό τεταρτημόριο της περιοχής 5, όπως φαίνεται, και το D στο άνω ή κάτω δεξιό τεταρτημόριο της περιοχής 6 (δεδομένου ότι η ακριβής θέση του C δεν έχει ακόμη καθοριστεί). Terminal propagation γίνεται με την τοποθέτηση δύο ειδών dummy cells – τοποθετημένα και εν μέρει τοποθετημένα σε κατάλληλες θέσεις. Έτσι, στην περιοχή 5, η επίδραση του A εκπροσωπείται από ένα dummy cell X κελί τοποθετημένο στο κάτω αριστερό τεταρτημόριο. Το dummy cell θα έλξη το B στο ίδιο τεταρτημόριο, ώστε να μειωθεί το net-cut. Στην περιοχή 6, το C cell που αντιπροσωπεύεται από μερικώς τοποθετημένο Y cell, που περιορίζεται στο άνω και κάτω δεξί τεταρτημόριο. Αυτό θα έλξη το D σε ένα από αυτά τα τεταρτημόρια.

Πληροφορίες ολικής δρομολόγησης χρησιμοποιούνται επίσης για τη βελτίωση της αποτελεσματικότητας της διάδοσης τερματικού. Για παράδειγμα, στο Σχήμα 20b, τα κελιά που συνδέονται στο ίδιο neto βρίσκονται και στα τέσσερα τεταρτημόρια και συνδέονται όπως φαίνεται. Εδώ, το A και το B θα πρέπει να επηρεάσουν ο ένας τη θέση άλλου μέσω Terminal propagation , και έτσι θα πρέπει να ισχύσει και για το B και C . Εφόσον δεν υπάρχει άμεση σύνδεση μεταξύ A και C, ωστόσο, δεν υπάρχει ανάγκη να τα διαδώσουμε (propagating). Μετά από κάθε βήμα διαμέρισης, τα κελιά σε διαφορετικά τεταρτημόρια συνδέονται σε ένα μοτίβο που δίνει το ελάχιστο κόστος, όπως συζητήθηκε παραπάνω. Όσο η διαμέριση προχωρά, αυτά τα μοτίβα σύνδεσης δίνουν ολικής δρομολόγησης δέντρο για κάθε neto. Κατά τη φάση του Terminal propagation , μόνον εκείνα τα modules που είναι άμεσα συνδεδεμένα μεταξύ τους πολλαπλασιάζονται. Τα βέλη δείχνουν την επίδραση του Terminal propagation. Για παράδειγμα, το κελί C θα πρέπει να ωθείται προς το άνω αριστερό τεταρτημόριο όταν η περιοχής 7 θα τεμαχιστεί σε 4 ταμάχια (quadrisectioned).

Ο αλγόριθμος έχει υλοποιηθεί ως μέρος του VPNR χώρου και του πακέτου διαδρομής. Προκαταρκτικά πειράματα δείχνουν ότι ο αλγόριθμος αυτός

συγκρίνεται ευνοϊκά σε σχέση με τον Timberwolf 3.2. Για διάφορα τυποποιημένα κυκλώματα κελιών, ο αλγόριθμος αυτός απέδωσε μια περιοχή στο $\pm 5\%$ της περιοχής που παράγεται από τον Timberwolf. Ο αλγόριθμος αυτός, εντούτοις, κατάφερε την ίδια ποιότητα διάταξης 50-200 φορές πιο γρήγορα από ό, τι ο Timberwolf. Ο χρόνος εκτέλεσης που αναφέρθηκε είναι της τάξης των 1.4 λεπτών για ένα 304-κελιών κυκλώμα και 1 ώρα για ένα 2907-κελιών κυκλώμα σε ένα VAX 8600.

4. Τοποθέτηση με τον γενετικό αλγόριθμο

Ο γενετικός αλγόριθμος είναι ένας πολύ ισχυρός αλγόριθμος βελτιστοποίησης, ο οποίος λειτουργεί με τη μίμηση της φυσικής διαδικασίας της εξέλιξης ως μέσο, προχωρώντας προς το βέλτιστο. Ιστορικά, προηγήθηκε της simulated annealing [Holland 1975], αλλά μόλις πρόσφατα έχει εφαρμοστεί ευρέως για την επίλυση προβλημάτων σε διάφορους τομείς, συμπεριλαμβανομένης της τοποθέτησης VLSI [Grefenstette 1985, 1987]. Ο αλγόριθμος ξεκινά με μια αρχική σειρά τυχαίων διαμορφώσεων, που ονομάζουμε ***πληθυσμός (population)***. Κάθε άτομο στον πληθυσμό είναι μια σειρά από σύμβολα, συνήθως μια δυαδική συμβολοσειρά που αντιπροσωπεύει μια λύση στο πρόβλημα βελτιστοποίησης. Κατά την διάρκεια κάθε επανάληψης, που ονομάζεται ***γενιά (generation)***, τα άτομα στον τρέχοντα πληθυσμό αξιολογούνται χρησιμοποιώντας το μέτρο της ***καταλληλότητας (fitness)***. Με βάση αυτή την τιμή της καταλληλότητας, ανεξάρτητα άτομα επιλέγονται από τον πληθυσμό δύο κάθε φορά, ως ***γονείς (parents)***. Οι πιο κατάλληλοι έχουν μια υψηλότερη πιθανότητα να επιλεγούν. Ένας αριθμός γενετικών φορέων εφαρμόζονται στους γονείς για τη δημιουργία νέων ατόμων, που ονομάζονται απόγονοι, συνδυάζοντας τα χαρακτηριστικά και των δύο γονέων. Οι τρεις γενετικοί φορείς που χρησιμοποιούνται συνήθως είναι οι: crossover, μετάλλαξη (mutation), και αναστροφή (inversion), τα οποία προέρχονται κατ 'αναλογία από τη βιολογική διαδικασία της εξέλιξης. Οι εν λόγω φορείς περιγράφονται με λεπτομέρειες παρακάτω. Στην συνέχεια, αξιολογούνται οι απόγονοι, και μια νέα γενιά σχηματίζεται με την επιλογή μερικών γονέων και απογόνων, και πάλι με βάση την καταλληλότητα τους, έτσι ώστε να κρατηθεί το μέγεθος του πληθυσμού σταθερό.

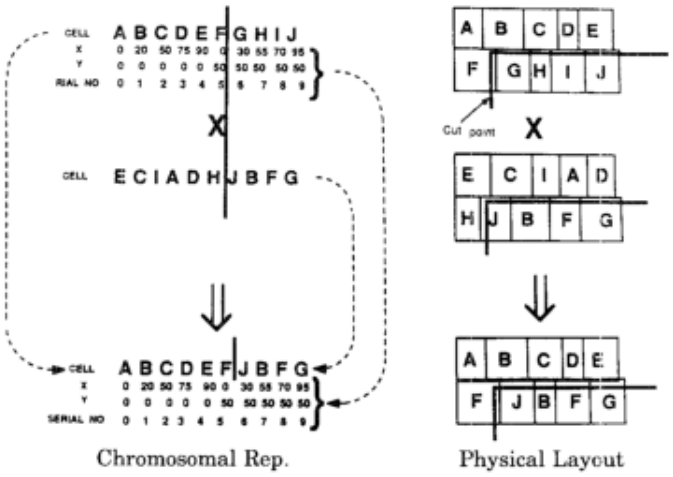
Αυτή η ενότητα εξηγεί γιατί οι γενετικοί αλγόριθμοι έχουν τόσο μεγάλη επιτυχία στο σύνθετο πρόβλημα βελτιστοποίησης όσον αφορά σχήματα και την επίδραση των γενετικών operators σε αυτά. Άτυπα, τα σύμβολα που χρησιμοποιούνται στις συμβολοσειρές των λύσεων είναι γνωστά ως **γονίδια (*genes)***. Αυτά είναι τα βασικά δομικά στοιχεία μιας λύσης και αντιπροσωπεύουν τις ιδιότητες που καθιστούν μία λύση διαφορετική από την άλλη. Για παράδειγμα, στο πρόβλημα τοποθέτησης των κελιών, οι ταξινομημένες τριπλέτες που αποτελούνται από τα κελιά και τις αντίστοιχες συντεταγμένες τους, μπορούν να θεωρηθούν ως *genes*. Μια συμβολοσειρά λύσεων, η οποία αποτελείται από τα γονίδια, λέγεται χρωμόσωμα. Ένα σχήμα είναι ένα σύνολο γονιδίων που συνθέτουν μια μερική λύση. Ένα παράδειγμα θα ήταν μια υπο-τοποθέτηση (subplacement), που αποτελείται από οποιονδήποτε αριθμό τέτοιων τριπλετών, μη λαμβάνοντας υπ'οψη τα υπόλοιπα τα κελιά. Ένα σχήμα με m στοιχεία καθορισμού και «δεν μας νοιάζει » για τις υπόλοιπες n-m θέσεις (όπως ένα m-cell subplacement σε ένα n-cell πρόβλημα τοποθέτησης) μπορεί να θεωρηθεί ως ένα (n - m) – διαστάσεων υπερεπίπεδο στη λύση για το χώρο. Όλα τα σημεία σε αυτό το υπερεπίπεδο (Δηλαδή, όλες οι διαμορφώσεις που περιέχουν το δεδομένο subplacement) είναι στιγμιότυπα του σχήματος. Σημείωση εδώ ότι το subplacement δεν χρειάζεται να γειτνιάζει φυσικά, όπως ένα ορθογώνιο patch της περιοχής του τσιπ. Για παράδειγμα, ένα καλό subplacement μπορεί να αποτελείται από δύο συνδεδεμένα πυκνά κελιά σε γειτονικές περιοχές. Παρομοίως, ένα καλό subplacement μπορεί επίσης να αποτελείται από ένα κελί στην άκρη της εισόδου του δικτύου και ένα κελί στη άκρη της εξόδου που είναι όμως τοποθετημένα στα αντίθετα

άκρα του τσιπ. Και τα δύο αυτά subplacements θα συμβάλουν στην υψηλή απόδοση του ατόμου που τα κληρονομεί. Έτσι, ένα σχήμα είναι μια λογική και όχι φυσική ομαδοποίηση των τριπλών κελιών-συντεταγμένων που έχουν μια συγκεκριμένη σχέση με την τοποθέτηση.

Όπως αναφέρθηκε παραπάνω, οι γενετικοί παράγοντες δημιουργούν μια νέα γενιά διαμορφώσεων συνδυάζοντας τα schemata (ή subplacements) από γονείς που επιλέγονται από την τρέχουσα γενιά. Λόγω της στοχαστικής διαδικασίας επιλογής, οι καταλληλότερη γονείς , οι οποίοι αναμένεται να περιέχουν κάποια καλα subplacements, είναι πιθανό να παράγουν περισσότερους απογόνους, και οι κακοί γονείς, οι οποίοι περιέχουν κάποια κακά subplacements, είναι πιθανό να παράγουν λιγότερους απογόνους. Έτσι, στην επόμενη γενιά, ο αριθμός των καλών subplacements (ή υψηλής καταλληλότητας schemata) τείνουν να αυξηθούν, και ο αριθμός των κακών subplacements (χαμηλής καταλληλότητας schemata) τείνουν να μειωθούν. Έτσι, η καταλληλότητα του συνόλου του πληθυσμού βελτιώνεται. Αυτός είναι ο βασικός μηχανισμός της βελτιστοποίησης από τον γενετικό αλγόριθμο.

Κάθε άτομο στον πληθυσμό είναι ένα παράδειγμα των 2^n σχημάτων, όπου n είναι το μήκος της κάθε ατομικής συμβολοσειράς. (Αυτό είναι ισοδύναμο με μια τοποθέτηση ένος n-κελιού που περιλαμβάνει 2^n subplacements οποιουδήποτε μεγέθους.) Έτσι, υπάρχει ένας πολύ μεγάλος αριθμός σχημάτων που εκπροσωπούν ένα σχετικά μικρό πληθυσμό. Δοκιμάζοντας έναν νέο απόγονο, έχουμε μια κατά προσέγγιση εκτίμηση της καταλληλότητας όλων των σχημάτων του ή των subplacements.

Εικόνα 28: Η παραδοσιακή μέθοδος του crossover.



Έτσι, με κάθε νέα διαμόρφωση που εξετάζεται , ο αριθμός του καθενός από 2^n σχήματα που παρουσιάζονται στον πληθυσμό ρυθμίζεται ανάλογα με την ακαταλληλότητά του. Αυτή η επίδραση έχει ονομαστεί ως εγγενής παραλληλισμός του γενετικού αλγορίθμου. Καθώς όλο και περισσότερες διαμορφώσεις δοκιμάζονται εκεί έξω, οι σχετικές αναλογίες των διαφόρων σχημάτων στον πληθυσμό αντανακλούν τη καταλληλότητα τους περισσότερο και με μεγαλύτερη ακρίβεια. Όταν ένα κατάλληλο σχήμα εισάγεται στον πληθυσμό μέσω ενός απογόνου, κληρονομείται από άλλους στη διάδοχη γενεά, συνεπώς και η αναλογία της στις αυξήσεις του πληθυσμού. Ξεκινά να οδήγηση στην έξοδο τα

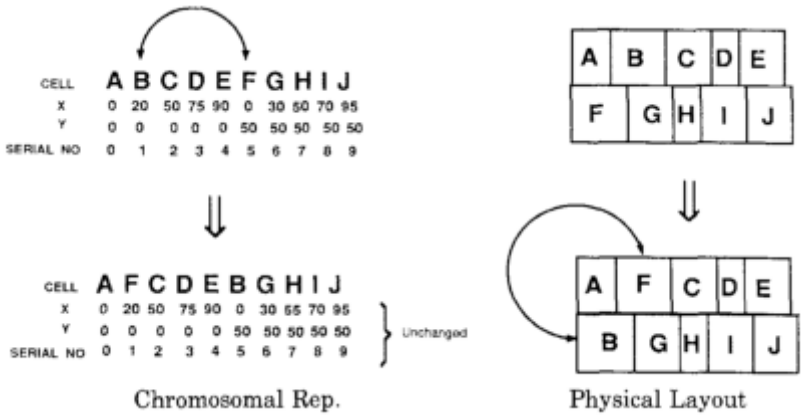
λιγότερο ταιριαστά σχήματα, και ο μέσος όρος της καταλληλότητας του πληθυσμού συνεχίζει να βελτιώνεται.

Οι γενετικοί παράγοντες και η σημασία τους μπορούν τώρα να εξηγηθούν.

Crossover. Ο Crossover είναι ο κύριος γενετικός operator(χειριστής). Λειτουργεί σε δύο άτομα ταυτόχρονα και παράγει απογόνους με το συνδυασμό σχημάτων από τους δύο γονείς. Ένας απλός τρόπος για να επιτύχουμε crossover θα είναι να επιλέξουμε ένα τυχαίο σημείο κοπής και να παράγουμε απογόνους με το συνδυασμό του τμήματος του ενός γονέα στα αριστερά του σημείου τομής με το τμήμα του άλλου γονέα στα δεξιά του σημείου κοπής. Αυτή η μέθοδος λειτουργεί καλά με τη παράσταση συμβολοσειράς bits. Το Σχήμα 28 δίνει ένα παράδειγμα του crossover. Σε ορισμένες εφαρμογές, όπου τα σύμβολα στο solution string δεν μπορούν να επαναληφθούν, αυτή η μέθοδος δεν μπορεί να εφαρμοστεί χωρίς τροποποιήσεις. Η Τοποθέτηση είναι ένα τυπικό πρόβλημα τομέα οπού αυτές οι συγκρούσεις μπορούν να συμβούν. Για παράδειγμα, όπως φαίνεται στο Σχήμα 28, τα κελιά B και F επαναλαμβάνονται, και τα κελιά H και I έχουν μείνει έξω. Επιπλέον, χρειαζόμαστε είτε ένα νέο crossover operator που λειτουργεί καλά για αυτούς τους τομείς του προβλήματος ή μια μέθοδο για την επίλυση αυτών των συγκρούσεων χωρίς να προκαλεί σημαντική υποβάθμιση στην απόδοση της διαδικασίας αναζήτησης. Η απόδοση του γενετικού αλγορίθμου εξαρτάται σε μεγάλο βαθμό από την απόδοση του crossover operator που χρησιμοποιείται. Διάφοροι crossover operators που υπερνικούν αυτά τα προβλήματα περιγράφονται στις ακόλουθες ενότητες.

Όταν ο αλγόριθμος έχει τρέξει για κάποιο χρονικό διάστημα, τα άτομα του πληθυσμού αναμένεται να είναι μετρίως καλά. Έτσι, όταν το σχήματα από δύο τέτοια άτομα έρχονται μαζί, το αποτέλεσμα στους απογόνους μπορεί να είναι ακόμη καλύτερο, όπου σε αυτή τη περίπτωση γίνονται αποδεκτοί στον πληθυσμό. Εκτός αυτού, οι κατάλληλοι γονείς έχουν μια μεγαλύτερη πιθανότητα δημιουργίας απογόνων. Αυτή η διαδικασία επιτρέπει στον αλγόριθμο να εξετάσει περισσότερες διαμορφώσεις σε μια περιοχή με μεγαλύτερη μέση καταλληλότητα, ώστε το βέλτιστο να μπορεί να προσδιοριστεί και, την ίδια στιγμή, να εξεταστούν μερικές διαμορφώσεις σε άλλες περιοχές της διαμόρφωσης χώρου, ώστε άλλες περιοχές με υψηλή απόδοση μέσο όρο να μπορούν να ανακαλυφθούν.

Εικόνα 29: Mutation



Η ποσότητα του crossover ελέγχεται από τον crossover rate(συντελεστής διασταύρωσης) , ο οποίος ορίζεται ως ο λόγος του αριθμού των απογόνων που παράγονται σε κάθε γενιά με το μέγεθος του πληθυσμού. Το crossover rate καθορίζει τον λόγο του αριθμού των αναζητήσεων σε περιοχές με μέση υψηλή καταλληλότητα με τον αριθμό των αναζητήσεων σε άλλες περιοχές. Ένα υψηλότερο crossover rate επιτρέπει την εξερεύνηση περισσότερου του χώρου λύσεων και μειώνει τις πιθανότητες της καθίζησης για ένα ψεύτικο βέλτιστο, αλλά αν αυτό το rate είναι πολύ υψηλό, αυτό οδηγεί σε μια σπατάλη του χρόνου υπολογισμού για την εξερεύνηση ελάχιστα ελπιδοφόρων περιοχών του χώρου λύσεων.

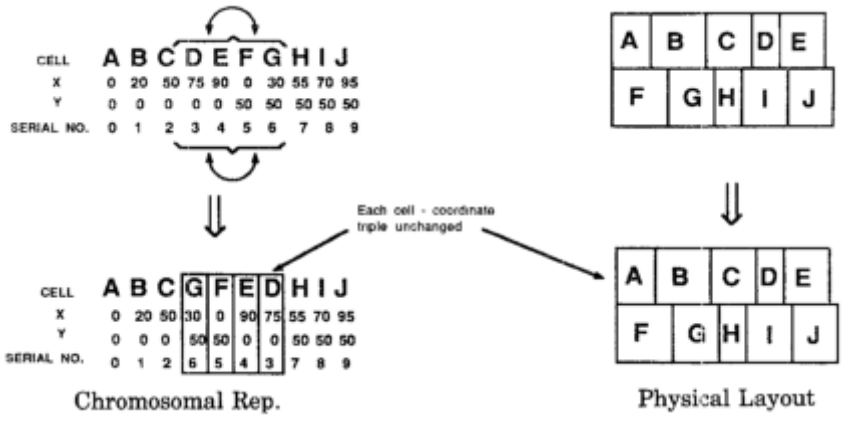
Mutation. Η μετάλλαξη (Mutation) είναι ένας background operator, ο οποίος δεν είναι άμεσα υπεύθυνος για την παραγωγή νέων απογόνων. Παράγει σταδιακά τυχαίες αλλαγές στους απογόνους που παράγονται από τον crossover. Ο μηχανισμός συνηθέστερα χρησιμοποιείται κατά ζεύγη ανταλλαγών όπως φαίνεται στο σχήμα 29. Αυτός δεν είναι ένας μηχανισμός για τυχαία εξέταση νέων διαμορφώσεων όπως και σε άλλους επαναληπτική βελτίωσης αλγορίθμους. Σε γενετικούς αλγόριθμους, η μετάλλαξη εξυπηρετεί το κρίσιμο ρόλο της αντικατάστασης των γονιδίων που χάνονται από τον πληθυσμό κατά την διάρκεια της διαδικασίας επιλογής, έτσι ώστε να μπορούν να δοκιμαστούν σε ένα νέο πλαίσιο ή να παρέχει τα γονίδια τα οποία δεν ήταν παρών στον αρχικό πληθυσμό. Δηλαδή όσον αφορά το πρόβλημα της τοποθέτησης, ένα γονίδιο που αποτελείται από ένα τριπλό έτοιμο κελί και των συναφών ιδανικών συντεταγμένες του, μπορεί να μην είναι παρόν σε οποιοδήποτε από τα άτομα στον πληθυσμό. (Δηλαδή, ότι συγκεκριμένα κελιά μπορεί να συνδέονται με μη-ιδανικές συντεταγμένες σε σχέση με όλα τα άλλα άτομα.) Σε αυτή περίπτωση, το crossover από μόνο του δεν θα βοηθήσει, διότι είναι μόνο ένας μηχανισμός κληρονομικότητας για τα υπάρχοντα γονίδια. Ο mutation operator παράγει νέες τριάδες κύτταρων-συντεταγμένες. Εάν οι νέες τριάδες συμπεριφέρονται καλά, οι διαμορφώσεις που περιέχουν αυτές διατηρούνται, και αυτές οι τριάδες θα εξαπλωθούν σε όλο τον πληθυσμό.

Ο ρυθμός μεταλλάξεως ορίζεται ως το ποσοστό του συνολικού αριθμού των γονιδίων του πληθυσμού, τα οποία είναι μεταλλαγμένα σε κάθε γενιά. Ετσι, για ένα n-cell πρόβλημα τοποθέτησης , με ένα μέγεθος πληθυσμού N_p, ο συνολικός αριθμός των γονιδίων είναι nN_p , και $nN_pR_m/2$ ζεύγη κόμβων που εκτελούνται για το mutation rate R_m . Το mutation rate ελέγχει το ρυθμό με τον οποίο εισάγονται νέα γονίδια

εντός του πληθυσμού για δοκιμή. Εάν είναι πολύ χαμηλό, τότε πολλά γονίδια που θα ήταν χρήσιμα ποτέ δεν δοκιμάστηκαν. Εάν είναι πάρα πολύ υψηλό, θα υπάρχει πάρα πολλές τυχαίες διαταραχές, οι απόγονοι θα αρχίσει να χάνουν την ομοιότητα τους με τους γονείς, και ο αλγόριθμος θα χάσει την ικανότητα να μαθαίνει από την ιστορία της αναζήτησης.

Inversion (Αναστροφή). Ο inversion operator παίρνει ένα τυχαίο τμήμα σε μια σειρά λύσεων και αντιστρέφει το τέλος για το τέλος

Εικόνα 30: Inversion



(Σχήμα 30). Η εργασία αυτή πραγματοποιείται κατά τέτοιο τρόπο ώστε να μην τροποποιήσει τη λύση που αντιπροσωπεύεται από τη συμβολοσειρά, αντ 'αυτού, τροποποιεί μόνο την αναπαράσταση της λύσης. Έτσι, τα σύμβολα που συνθέτουν τη συμβολοσειρά πρέπει να έχει μια ερμηνεία ανεξάρτητη από την θέσης τους. Αυτό μπορεί να επιτευχθεί συνδέοντας ένα σειριακό αριθμό με κάθε σύμβολο στη συμβολοσειρά και ερμηνεύοντας τη συμβολοσειρά με σεβασμό σε αυτούς τους σειριακούς αριθμούς αντί του δείκτη του πίνακα. Όταν ένα σύμβολο κινείται στο πίνακα, ο σειριακός αριθμός κινείται μαζί του, έτσι ώστε η ερμηνεία του συμβόλου να παραμένει αμετάβλητη. Στην πρόβλημα τοποθέτησης κυψελών , η x-και y-συντεταγμένες που αποθηκεύονται με κάθε κελί εκτελούν αυτή τη λειτουργία. Έτσι, δεν έχει σημασία που το τριπλό κελί-συντεταγμένη βρίσκεται στο πίνακα του πληθυσμού, θα έχει την ίδια ερμηνεία με αυτη της φυσικής διάταξης.

Το πλεονέκτημα του inversion operator είναι το ακόλουθο. Υπάρχουν κάποιες ομάδες ιδιοτήτων, ή γονίδιων , τα οποία θα είναι επωφελής για τους απογόνους να κληρονομήσουν μαζί από τον ένα γονέα. Τέτοιες ομάδες γονιδίων, οι οποίες αλληλεπιδρούν για να αυξηθεί η ικανότητα των απογόνων τους που κληρονομούν, ονομάζεται coadapted. Για παράδειγμα, αν τα κελιά A και B είναι στενά συνδεδεμένα μεταξύ τους και ο γονέας 1 έχει τα γονίδια (A, x_1, y_1) και (B, x_2, y_2) όπου (x_1, y_1) και (x_2, y_2) είναι γειτονικές περιοχές, θα είναι πλεονεκτικό για τους απογόνους να κληρονομήσει και τα δύο αυτά γονίδια από ένα γονέα ώστε μετά τα κελιά crossover A και B να παραμείνουν σε γειτονικές περιοχές. Αν δύο γονίδια είναι κοντά στη γραμμή λύσεων, έχουν μικρότερη πιθανότητα να χωρίσουν όταν ο inversion operator χωρίσει το string σε δύο τμήματα. Έτσι, με το ανακάτεμα των κελιών γύρω από τη γραμμή λύσεων, η αναστροφή επιτρέπει τριάδες των κελιών που είναι ήδη καλά τοποθετημένες σε σχέση το ένα προς το άλλο να βρίσκεται κοντά το ένα στο άλλο

στο string. Αυτό αυξάνει την πιθανότητα ότι, όταν ο χειριστής crossover χωρίζει διαμορφώσεις γονέων σε τμήματα για να περάσει στους απογόνους, τα subplacements που αποτελούνται από τέτοιες ομάδες θα περάσουν ανέπαφα από τον ένα γονέα (ή άλλο). Αυτή η μέθοδος επιτρέπει τον σχηματισμό και την επιβίωση των ιδιαίτερα βελτιστοποιημένων subplacements πολύ πριν από την βελτιστοποίηση της κάθε ολοκληρωμένη τοποθέτηση όταν αυτή έχει ολοκληρωθεί. Ο inversion rate είναι η πιθανότητα της εκτέλεσης αναστροφής σε κάθε άτομο κατά τη διάρκεια κάθε γενεάς. Ελέγχει την ποσότητα του σχηματισμού μιας ομάδας. Πάρα πολύ αναστροφή θα έχει ως αποτέλεσμα τη διατάραξη των ομάδων που έχουν ήδη σχηματιστεί.

Selection (Επιλογή). Μετά τη δημιουργία απογόνων, χρειαζόμαστε μια διαδικασία επιλογής για να επιλέξουμε την επόμενη γενιάς από το συνδυασμό σύνολο των γονέων και των απογόνων. Υπάρχει μεγάλη διαφορετικότητα στις λειτουργίες επιλογής που χρησιμοποιούνται από διάφορους ερευνητές. Αυτή η ενότητα συνοπτικά παραθέτει κάποιες από αυτές. Οι ακόλουθες ενότητες δίνουν τις συγκεκριμένες λειτουργίες που χρησιμοποιούνται κυρίως στους αλγορίθμους. Οι τρεις μέθοδοι επιλογής που χρησιμοποιούνται πιο συχνά είναι ο competitive(ανταγωνιστικός), random (τυχαίος), και ο stochastic(στοχαστικός).

Στην competitive selection(ανταγωνιστική επιλογή), όλοι οι γονείς και οι απόγονοι ανταγωνίζονται μεταξύ τους, και τα P καταλληλότερα άτομα επιλέγονται, όπου P είναι το μέγεθος του πληθυσμού.

Στην τυχαία επιλογή, όπως υποδηλώνει το όνομα, τα άτομα P επιλέγονται τυχαία, με ομοιόμορφη πιθανότητα. Μερικές φορές αυτό είναι πλεονεκτικό, γιατί με αυτόν τον τρόπο ο πληθυσμός διατηρεί την ποικιλομορφία του πολύ περισσότερο και η αναζήτηση δεν συγκλίνει σε ένα τοπικό βέλτιστο. Με καθαρό competitive selection, το σύνολο του πληθυσμού μπορεί γρήγορα να συγκλίνουν σε άτομα που είναι ελαφρώς διαφορετικά από τα άλλα, και μετά ο αλγόριθμος θα χάσει την ικανότητά του να βελτιστοποιήσει περαιτέρω. (Η κατάσταση αυτή ονομάζεται πρόωρη σύγκλιση. Μόλις συμβεί αυτό, ο πληθυσμός θα πάρει πολύ χρόνο για να ανακτήσει την ποικιλομορφία του μέσα από την αργή διαδικασία της μετάλλαξης.) Μία παραλλαγή αυτής της μεθόδου είναι η διατήρηση της καλύτερης διαμόρφωσης και η επιλογή του υπόλοιπου του πληθυσμού τυχαία. Αυτό εξασφαλίζει ότι η καταλληλότητα θα αυξάνει πάντα μονότονα και εμείς δεν πρόκειται ποτέ να χάσουμε την καλύτερη διαμόρφωση που έχουμε ήδη βρει, απλά επειδή δεν επιλέχθηκε από την διαδικασία με τη τυχαία επιλογή.

Η Στοχαστική επιλογή είναι παρόμοιο με τη διαδικασία που περιγράφηκε παραπάνω για την επιλογή των γονέων για το crossover. Η πιθανότητα επιλογής του κάθε ατόμου είναι ανάλογη με την καταλληλότητά του. Αυτή η μέθοδος περιλαμβάνει τόσο τον ανταγωνισμό όσο και τύχη.

Comparison with Simulated Annealing . Τόσο το simulated annealing όσο ο γενετικός αλγόριθμος είναι υπολογιστικά εντατικοί .Ο γενετικός αλγόριθμος, όμως, έχει ορισμένες ενσωματωμένες λειτουργίες, οι οποίες, αν αξιοποιηθούν σωστά, μπορεί να οδηγήσουν σε σημαντική εξοικονόμηση. Μία διαφορά είναι ότι το simulated annealing λειτουργεί μόνο σε μία διαμόρφωση σε μια μονάδα του χρόνου,

ενώ ο γενετικός αλγόριθμος διατηρεί ένα μεγάλο μέρος του πληθυσμού των διαμορφώσεων που έχουν βελτιστοποιηθεί ταυτόχρονα. Έτσι, ο γενετικός αλγόριθμος επωφελείται από την εμπειρία που αποκτά στο παρελθόν από την εξερεύνηση για λύση του χώρου και μπορεί να κατευθύνει μια πιο εκτεταμένη έρευνα σε τομείς με χαμηλή μέσο κόστος. Δεδομένου ότι το simulated annealing λειτουργεί μόνο σε μία διαμόρφωση κάθε φορά, έχει μικρή ιστορία να χρησιμοποιήσει για να μάθει από το δοκιμές στο παρελθόν.

Τόσο το simulated annealing όσο και ο γενετικός αλγόριθμος διαθέτουν μηχανισμούς για την αποφυγή παγίδευσης σε τοπικά βέλτιστα. Στο simulated annealing, αυτό γίνεται απορρίπτοντας περιστασιακά μια ανώτερη διαμόρφωση και αποδεχόμενος μια κατώτερη .Ο γενετικός αλγόριθμος επίσης, βασίζεται σε κατώτερες διαμορφώσεις ως ένα μέσο αποφυγής εσφαλμένων βέλτιστων, αλλά δεδομένου ότι έχει ένα ολόκληρο πληθυσμό από διαμορφώσεις, ο γενετικός αλγόριθμος μπορεί να κρατήσει και να επεξεργαστεί κατώτερα διαμορφώσεις χωρίς να χάσει τις καλύτερες εξ αυτών. Εξάλλου, στο γενετικό αλγόριθμο κάθε νέα διαμόρφωση είναι κατασκευασμένη από δύο προηγούμενες διαμορφώσεις, πράγμα που σημαίνει ότι σε λίγες επαναλήψεις, όλες οι διαμορφώσεις στον πληθυσμό έχουν την ευκαιρία να συμβάλλουν με τα καλά χαρακτηριστικά τους για να σχηματίσουν μια superconfiguration. Στο simulated annealing, κάθε νέα ρύθμιση σχηματίζεται από μια μόνο παλιά διαμόρφωση, πράγμα που σημαίνει ότι τα καλά χαρακτηριστικά των περισσότερων σε μία δραστική διαφορετική διαμόρφωση ποτέ δεν αναμειγνύονται. Μια διαμόρφωση είτε γίνεται αποδεκτή ή απορρίπτεται ως σύνολο, ανάλογα με το συνολικό κόστος του.

Από την αρνητική πλευρά, ο γενετικός αλγόριθμος απαιτεί περισσότερο χώρο μνήμης σε σύγκριση με τον simulated annealing . Για παράδειγμα, σε ένα πρόβλημα 1000 τοποθετήσεως κυψελών θα απαιτούσε μέχρι 400kb για να αποθηκεύσει ένα πληθυσμό των 50 διαμορφώσεων . Για μετρίου μεγέθους προβλήματα , αυτή η απαίτηση μνήμης δεν μπορεί να αποτελέσει σημαντικό πρόβλημα, διότι εμπορικοί σταθμών εργασίας έχουν 4MB ή περισσότερο κύρια μνήμη. Για τα κυκλώματα της τάξεως των 10000 κελιών, ο γενετικός αλγόριθμος αναμένεται να έχει μία μικρή ποσότητα επιπλέον επιβάρυνση σε σελιδοποίηση σε σύγκριση με το simulated annealing , αλλά εξακολουθεί να επιταχύνει τη βελτιστοποίηση που οφείλεται στην αποτελεσματικότητα της διαδικασίας αναζήτησης.

Ο γενετικός αλγόριθμος είναι μια νέα και ισχυρή τεχνική. Η επιτυχία της μεθόδου αυτής εξαρτάται από την κατάλληλη επιλογή των διαφόρων παραμέτρων και συναρτήσεων που ελέγχουν διαδικασίες όπως μετάλλαξη, επιλογή, και crossover. Εάν οι συναρτήσεις επιλεχθούν σωστά, θα έχουμε μια καλή τοποθέτηση. Το σημαντικότερο πρόβλημα στον σχεδιασμό ενός γενετικού αλγορίθμου για την τοποθέτηση modules είναι η επιλογή των πιο κατάλληλων συναρτήσεων για το πρόβλημα αυτό. Ένα μεγάλο μέρος της έρευνας σήμερα διεξάγεται σε αυτό. Στην ενότητα αυτή, τρεις αλγόριθμοι συζητούνται, οι Cohoon και Paris [1986], ο Kling [1987], και ο Shahookar και hkzumder [1990].Περισσότερη δουλειά έχει γίνει σε αυτόν τον τομέα από τον Chan και Mazumder [1989]

4.1 Genie: Genetic Placement Algorithm

Ο αλγόριθμος Genie δημιουργήθηκε από τους Cohoon και Paris [1986]. Ο ψευδοκώδικας δίνεται παρακάτω:

```
PROCEDURE Genie:
  Initialize;
  N_P ← population size;
  N_O ← N_p P_ψ;
    /* where P_ψ is the desired ratio of the number of offspring to the population size */
  Construct_population(N_P);
  FOR i ← 1 TO N_P score(population[i]);
  ENDFOR;
  FOR i ← 1 TO Number_of_Generations
    FOR j ← 1 TO N_O
      (x, y) ← Choose_Parents;
      offspring[j] ← generate_Offspring(x, y);
      Score(offspring[j]);
    ENDFOR;
    population ← Select(population, offspring, N_P);
    FOR j ← 1 TO N_P
      With probability P_μ Mutate(population[j]);
    ENDFOR;
  ENDFOR;
  Return highest scoring configuration in population;
END.
```

Τα ακόλουθα είναι μια περιγραφή μερικών από τις λειτουργίες που χρησιμοποιούνται στο Cohoon και Paris [1986] και τα αποτελέσματα τους.

(1) Η αρχική κατασκευή του πληθυσμού.

Ο Cohoon και Paris [1986] πρότειναν δύο μεθόδους για τη δημιουργία του αρχικού πληθυσμού. Ο πρώτος είναι να κατασκευάστει ο πληθυσμός τυχαία. Ο δεύτερος είναι να χρησιμοποιήσουμε μια άπληστη τεχνική συσταδοποίησης για να τοποθετήσουμε τα κελιά. Ένα net επιλέγεται τυχαία, και οι μονάδες συνδέονται με αυτό και τοποθετούνται σε μια netlist σειρά. Στη συνέχεια, ένα άλλο neto συνδεδεμένο με την πιο πρόσφατα τοποθετημένη μονάδα επιλέγεται, και η διαδικασία επαναλαμβάνεται. Πειραματικές παρατηρήσεις δείχνουν ότι ο αρχικός πληθυσμός κατασκευασμένο από ομαδοποίηση είναι καταλληλότερος, αλλά συγκλίνει γρήγορα σε ένα τοπικό βέλτιστο. Ως εκ τούτου, στον τελικό αλγόριθμο, έχουν χρησιμοποιήσει ένα μικτό πληθυσμό, ένα μέρος των οποίων είναι κατασκευασμένος από κάθε μέθοδο.

(2) Συνάρτηση Βαθμολόγησης.

Η συνάρτηση βαθμολόγησης καθορίζει την καταλληλότητα της τοποθέτησης. Η συνάρτηση βαθμολόγησης σ που χρησιμοποιείται στη Genie χρησιμοποιεί τη συμβατική συνάρτηση μήκους καλωδίου με βάση την οριοθέτηση ορθογωνίου. Δεν ενδιαφέρεται για επικάλυψη κελιών ή μήκη γραμμών, λόγω του στυλ διάταξης του πίνακα πυλών. Ωστόσο, ενδιαφέρεται η χρήση του καναλιού να είναι ανομοιόμορφη. Αυτό γίνεται ως εξής:

Έστω $\boldsymbol{L_i}$, να είναι η περίμετρος του δικτύου i, r και c είναι ο αριθμό των γραμμών και στηλών, αντίστοιχα, h_i είναι ο αριθμός των δικτύων που τέμνουν το οριζόντιο κανάλι $\boldsymbol{i}$, v_i είναι ο αριθμός των δικτύων που τέμνουν το κάθετο κανάλι $\boldsymbol{i}$, s_h είναι η τυπική απόκλιση του h_i, s_v είναι η τυπική απόκλιση του v_i , $\bar{h}$ και $\bar{v}$ είναι η μέση τιμή του h_i, και του v_i, αντίστοιχα,

$$\hat{h_i} = \begin{cases} h_i - \bar{h} - s_h & \text{if} \quad \bar{h} + s_h < h_i \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{v_i} = \begin{cases} v_i - \bar{v} - s_v & \text{if} \quad \bar{v} + s_v < v_i \\ 0 & \text{otherwise} \end{cases}$$

Τότε

$$\sigma = \frac{1}{2}\sum_{i=1}^{n} L_i + \sum_{i=1}^{r-1} \hat{h}_i^2 + \sum_{i=1}^{c-1} \hat{v}_i^2.$$

Αυτή η συνάρτηση βαθμολόγησης τιμωρεί όλα τα κανάλια που έχουν μια πυκνότητα στη καλωδίωση περισσότερο από μία τυπική απόκλιση πάνω από την μέση. Έτσι, ενθαρρύνει μια πιο ομοιόμορφη κατανομή της καλωδίωσης.

(3) Parent choice function.

Η συνάρτηση επιλογής γονέα επιλέγει τα ζευγάρια γονέων. Τέσσερις εναλλακτικές λύσεις έχουμε εδώ. (1) ζεύγος μια τυχαίας σειράς με το καταλληλότερο αλφαριθμητικό, (2) να επιλέξουν τους δύο γονείς τυχαία, (3) επιλέξτε γονείς στοχαστικά, όπου η πιθανότητα του κάθε ατόμου που επιλέγεται ως γονέας είναι ανάλογη με την καταλληλότητά του, και (4), το οποίο είναι το ίδιο με (3), αλλά επιτρέπει μόνο τα άτομα με άνω του μέσου όρου καταλληλότητας να αναπαραχθούν. Η συνάρτηση καταλληλότητας που χρησιμοποιείται εδώ είναι

$$w(s) = \frac{\sigma_{\max}}{\sigma(s)},$$

η οποία είναι ισοδύναμη για τη λήψη του αμοιβαίου κόστους και για την εξομάλυνση έτσι ώστε η χαμηλότερη καταλληλότητα να είναι 1. Αν η καλύτερη διαμόρφωση συνδυάζεται με μια τυχαία, ο πληθυσμός χάνει γρήγορα την ποικιλομορφία του και ο αλγόριθμος συγκλίνει σε ένα φτωχό τοπικό ελάχιστο. Στο άλλο άκρο, αν οι γονείς επιλέγονται τυχαία, υπάρχει μικρή βελτίωση μετά από αρκετές γενιές και ως εκ τούτου δεν υπάρχει σύγκλιση σε μια καλή τοποθέτηση. Η στοχαστικές συναρτήσεις (3) και (4) παράγουν τα καλύτερα αποτελέσματα.

(4) Crossover Operator.

Ο χειριστής crossover παράγει απογόνους από τους γονείς. Δύο crossover operators έχουν ήδη περιγραφεί. Ο πρώτος επιλέγει ένα τυχαίο module e_s και φέρνει τους

τέσσερις κοντινότερους γείτονές στον γονέα 1 σε γειτονικές υποδοχές στον γονέα 2 ενώ φέρνει σε αταξία ελάχιστα τις άλλες ενότητες. Για να το κάνετε αυτό, οι ενότητες που καταλαμβάνουν τις γειτονικές θέσεις του e_s στον γονέα 2 μετατοπίζονται προς τα έξω μία θέση ανά μονάδα χρόνου σε μια κίνηση αλυσίδας έως ότου μια κενή θέση βρεθεί (Σχήμα 31α). Το αποτέλεσμα αυτού είναι ότι το patch που αποτελείται από τις ενότητας e_s και τους τέσσερις γείτονες αντιγράφεται από τον γονέα 1 στον γονέας 2, και οι άλλες ενότητες μετατοπίζονται κατά μια θέση προκειμένου να δημιουργηθεί χώρος. Ο δεύτερος χειριστής επιλέγει ένα τετράγωνο που αποτελείται από ενότητες KxK από τον γονέα 1, όπου k είναι ένας τυχαίος αριθμός με μέση διακύμανση 3 και 1, και το αντιγράφει στο γονέα 2. Αυτή η μέθοδος τείνει να επαναλάβει κάποιες μονάδες και να αφήσει έξω κάποιες άλλες. Για να αποφευχθεί αυτή η σύγκρουση, οι ενότητες που βρίσκονται στο τετράγωνο patch του γονέα 2, αλλά όχι στο patch του γονέα 1, και οι οποίες πρόκειται να αντικατασταθούν αντιγράφονται στους χώρους των μονάδων που είναι στο patch του γονέα 1 αλλά όχι στο patch του γονέα 2, το οποίο κατά συνέπεια εμποδίζεται από το να γίνει dublicated (Σχήμα 31b). Πειράματα ευνοούν τον δεύτερο operator.

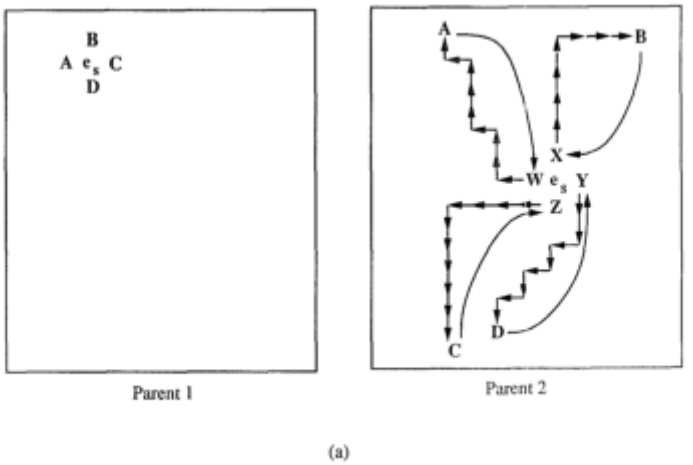
(5) Συνάρτηση επιλογής.

Η συνάρτηση επιλογής καθορίζει ποιες διαμορφώσεις θα επιβιώσουν στην επόμενη γενιά. Οι τρεις λειτουργίες που έχουν δοκιμάσει είναι (1) να επιλέξετε την καλύτερη σειρά και p -1 τυχαίες συμβολοσειρές για την επιβίωσή τους στην επόμενη γενιά, όπου p είναι το μέγεθος του πληθυσμού, (2) να επιλεγούν τυχαία p συμβολοσειρές, (3) να επιλέξουμε συμβολοσειρές στοχαστικά, με την πιθανότητα της επιλογή τους να είναι ανάλογη προς τη καταλληλότητα. Τα αποτελέσματα είναι παρόμοια με εκείνα της συνάρτησης επιλογής γονέα. Αν έχουν επιλεγεί η καλύτερη διαμόρφωση και p - 1 τυχαίες διαμορφώσεις, ο πληθυσμός χάνει γρήγορα την ποικιλομορφία του και συγκλίνει σε ένα φτωχό τοπικό ελάχιστο. Η συνάρτηση που επιλέγει κάθε διαμόρφωση p, τυχαία ή αυτή που πιθανοτικά ευνοεί την επιλογή των υψηλότερων σε βαθμολόγηση διαμορφώσεων αποδίδουν πολύ καλύτερα.

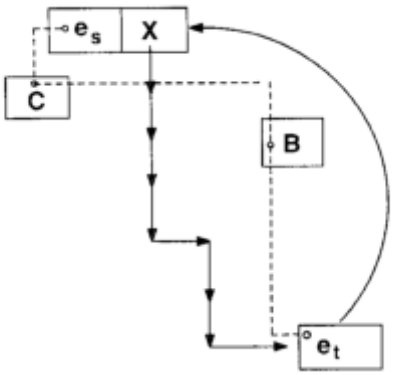
(6) Mutation Function (Συνάρτηση μετάλλαξης).

Δύο εναλλακτικές λύσεις (1) Εκτέλεση μια σειράς τυχαίων εναλλαγών και (2) χρησιμοποιήστε μια άπληστη τεχνική για τη βελτίωση του κόστος τοποθέτησης. Ο άπληστος operator επιλέγει ένα module e_s , σε neto Z και ψάχνει για ένα module e_t στο ίδιο neto που είναι το πλέον απομακρυσμένο από το e_s. Το e_t κατόπιν έρχεται κοντά στο e_s, και τα εκτοπισμένα modules θα μετατοπιστούν ένα διάστημα κάθε φορά μέχρις ότου μια κενή θέση βρεθεί (Σχήμα 32). Έτσι, η περίμετρος της οριοθέτησης ορθογωνίου του δικτύου Z μειώνεται παράλληλα μειώνεται ελάχιστα και το υπόλοιπο της τοποθέτησης.

Εικόνα 31: Crossover σε Ginie



Εικόνα 32: Greedy mutation



Πειραματικά Αποτελέσματα.

Η συγκριτική απόδοση των διαφορετικών παραλλαγών των γενετικών operators έχει περιγραφεί παραπάνω. Ο αλγόριθμος δοκιμάστηκε σε πέντε κυκλώματα με 36-81 κελιά. Η απόδοση συγκρίθηκε έναντι ενός αλγόριθμου simulated annealing επίσης κατασκευασμένο από Cohoon and Paris. Ο αλγόριθμος Genie λαμβάνει την ίδια ποιότητα τοποθέτησης σε δύο περιπτώσεις και έως 7% χειρότερη στις υπόλοιπες τρεις περιπτώσεις. Ο αριθμός των διαμορφώσεων που εξετάστηκαν, ωστόσο, ήταν μόνο 28% για ένα κύκλωμα, 50% για δύο κυκλώματα, και 75% για δύο κυκλώματα. Ο πραγματικός χρόνος της CPU δεν δόθηκε.

4.2 ESP: Evolution-Based Placement Algorithm

Ο Kling [1987] και οι Kling και Bannerjee [1987] ανέπτυξαν έναν evolution-based αλγόριθμο που χρησιμοποιεί επαναληπτικά την ακολουθία μετάλλαξης, την αξιολόγηση, την κρίση, και την ανακατανομή.Ο αλγόριθμος λειτουργεί μόνο σε μία διαμόρφωση κάθε στιγμή. Οι ενότητες στη διαμόρφωση αντιμετωπίζονται ως ο πληθυσμός. Μια μετάλλαξη είναι μια τυχαία αλλαγή στην τοποθέτηση. Η εκτίμηση καθορίζει το goodness της τοποθέτησης της κάθε ενότητας, δηλαδή, η ατομική συνεισφορά της μονάδας προς το κόστος. Ο Kling χρησιμοποιήσει αυτό το μέτρο της καλοσύνης αντί της παραδοσιακής καταλληλότητας στους γενετικούς αλγόριθμους. Η συνάρτηση κρίσεων (judgment function) πιθανοτικά καθορίζει αν μια μονάδα πρέπει να αφαιρεθεί και εκ νέου να ξανατοποθετηθεί ή όχι βάσει της αξίας του goodness. Στη φάση της ανακατανομής, όλες οι ενότητες που αφαιρείται κατά την judgment φάση τοποθετούνται σε νέες περιοχές. Οι αλγόριθμοι που χρησιμοποιούνται για την εκτέλεση αυτών των συναρτήσεων περιγράφονται λεπτομερώς παρακάτω.

Μετάλλαξη. Η Μετάλλαξη γίνεται με τυχαία εναλλαγή ενός ορισμένου αριθμού ζευγών modules χωρίς να λαμβάνεται υπόψη η επίδρασή τους στην τοποθέτηση. Η διεργασία μετάλλαξης ελέγχεται από δύο παραμέτρους που παρέχονται από το χρήστη -την πιθανότητα εμφάνισης της μετάλλαξης και το ποσοστό του συνολικού αριθμού των ενοτήτων που πρόκειται να μεταλλαχθούν. Αυτές οι δύο παράμετροι καθορίζουν τον αριθμό των μεταλλάξεων που εκτελούνται κατά τη διάρκεια κάθε επανάληψης.

Αξιολόγηση. Η αξιολόγηση είναι μια σύνθετη διαδικασία και είναι το κρίσιμο βήμα σε αυτόν τον αλγόριθμο. Όπως αναφέρθηκε παραπάνω, καθορίζει το goodness της τοποθέτησης της κάθε ενότητα, έτσι ώστε οι φτωχά τοποθετούμενες ενότητες μπορούν να αφαιρεθούν για την κατανομή. Ο Kling έχει προτείνει διάφορες διαδικασίες για την αξιολόγηση της αξία της goodness.

Για πίνακα πυλών, η goodness κάθε μονάδας υπολογίζεται ως ο λόγος της τρέχουσας τιμής με την προ-υπολογιστική ιδανική τιμή. Η εκτίμηση της τρέχουσας τιμής βασίζεται στο προϊόν της σύνδεσης με άλλες μονάδες και την αμοιβαία απόσταση από αυτούς. Ένα παράθυρο αξιολόγησης αποτελούμενο από τις κανονικοποιημένες αμοιβαίες αποστάσεις Μανχάταν από το κέντρο (που ονομάζονται βάρη) είναι precomputed όπως φαίνεται στο Σχήμα 33α. Για να αξιολογήσουμε την τρέχουσα τιμή μιας μονάδας i, το παράθυρο αξιολόγησης

επικεντρώνεται πάνω από αυτό. Για όλα τα j modules με τα οποία είναι συνδεδεμένο, το άθροισμα

$$r_i = \sum c_{ij} w_j,$$

υπολογίζεται, όπου c_{ij} , είναι η συνδεσιμότητα της μονάδας που αξιολογείται στο j-οστό module και w_j είναι το βάρος που αντιστοιχεί στην απόσταση. Το Σχήμα 33β δείχνει ένα παράδειγμα του υπολογισμού της τρέχουσας τιμής για ένα module . Η προ – υπολογιστική ιδανική τιμή λαμβάνεται με μια παρόμοια διαδικασία υπολογισμού, αλλά εδώ όλες οι ενότητες που συνδέονται στο module που αξιολογείται υποτίθεται ότι τοποθετούνται στην άμεση γειτονιά του, έτσι ώστε οι μονάδες με τη μεγαλύτερη συνδεσιμότητα να τοποθετούνται πλησιέστερα προς αυτό (Σχήμα 33c). Αυτό είναι το άνω όριο για την τρέχουσα τιμή, η οποία θα μπορούσε να επιτευχθεί μόνο από τα καλύτερα τοποθετημένα modules , τα οποία έχουν όλα συνδεδεμένες ενότητες σε παρακείμενες θέσεις.

Για τα standard cells, τρεις μέθοδοι έχουν προταθεί. Στην πρώτη, η ομόκεντρη μέθοδος κύκλου (Σχήμα 34), η περιοχή των ενοτήτων που συνδέονται με μονάδα i υπολογίζεται. Οι ομόκεντροι κύκλοι που ορίζονται στη συνέχεια έτσι ώστε ο νιοστός κύκλος καλύπτει n φορές αυτή την περιοχή. Βάρη έχουν εκχωρηθεί στους κύκλους από 100% για τον εσωτερικό κύκλο και 0% για την περιοχή έξω από τον εξωτερικό κύκλο. Η τρέχουσα τιμή ενός κελιού i προσδιορίζεται από το άθροισμα

$$r_i = \sum c_{ij} w_j,$$

Όπου c_{ij} είναι η συνδεσιμότητα με το j-οστό cell και w_j είναι το βάρος της κυκλικής περιοχής στην οποία βρίσκεται το pin του j-οστού κελιού.

Η δεύτερη μέθοδος αξιολόγησης για τα standard cells βασίζεται στο ελάχιστο δυνατό ορθογώνιο οριοθέτησης για ένα δίχτυ. Το ελάχιστο ορθογώνιο για κάθε δίχτυ υπολογίζεται με την τοποθέτηση όλων των ενοτήτων που συνδέονται με το εν λόγω δίχτυ στις πλησιέστερα γειτονικές περιοχές. Για να υπολογίσετε την goodness τιμή ενός τοποθετημένου module , η απόσταση του από το κέντρο βάρους του δικτύου υπολογίζεται. Εάν βρίσκεται εντός του ελαχίστου ορθογωνίου οριοθέτησης, η τιμή goodness είναι στο 100% . Διαφορετικά, είναι ο λόγος του ορίου του βέλτιστου ορθογωνίου του δικτύου με τις συντεταγμένες του κελιού που βρίσκονται πλησιέστερα προς κέντρο του δικτύου .

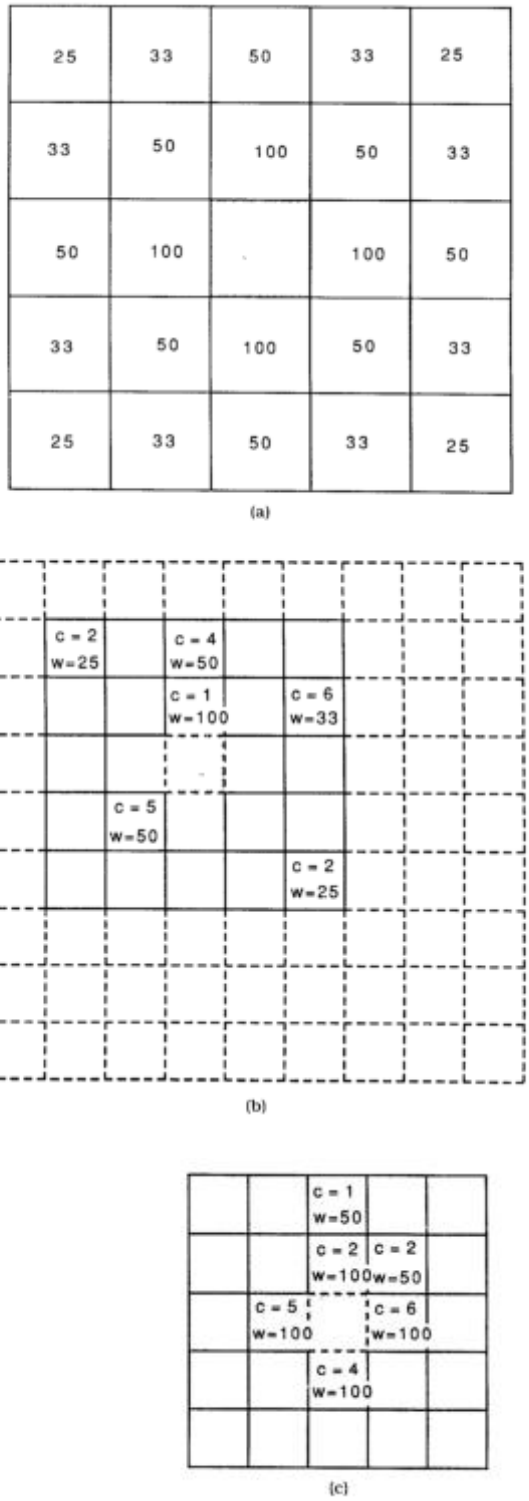
Η Τρίτη μέθοδος για standard cells βασίζεται στον λόγο του τρέχοντος μήκους σύρματος των δικτύων που συνδέονται σε ένα κελί με τα αντίστοιχα βέλτιστα μήκη καλωδίων. Το goodness υπολογίζεται από το μέσο όρο αυτής της αναλογίας για όλα τα neta που συνδέονται με το κελί που αξιολογείται. Το αποτέλεσμα στη συνέχεια κανονικοποιείται στο εύρος 0-100%. Αυτή η διαδικασία δίνει τα καλύτερα αποτελέσματα για standard cells.

Judgment (Κρίση). Στη φάση της κρίσης, modules που βρίσκονται σε κακή θέση αφαιρούνται για ανακατανομή. Η πιθανότητα απομάκρυνσης μιας ενότητας αυξάνεται όσο το goodness του μειώνεται. Το goodness της κάθε μονάδας συγκρίνεται σε σχέση με έναν τυχαίο αριθμό. Εάν το goodness είναι μικρότερο από τον τυχαίο αριθμό, αυτό απομακρύνεται.

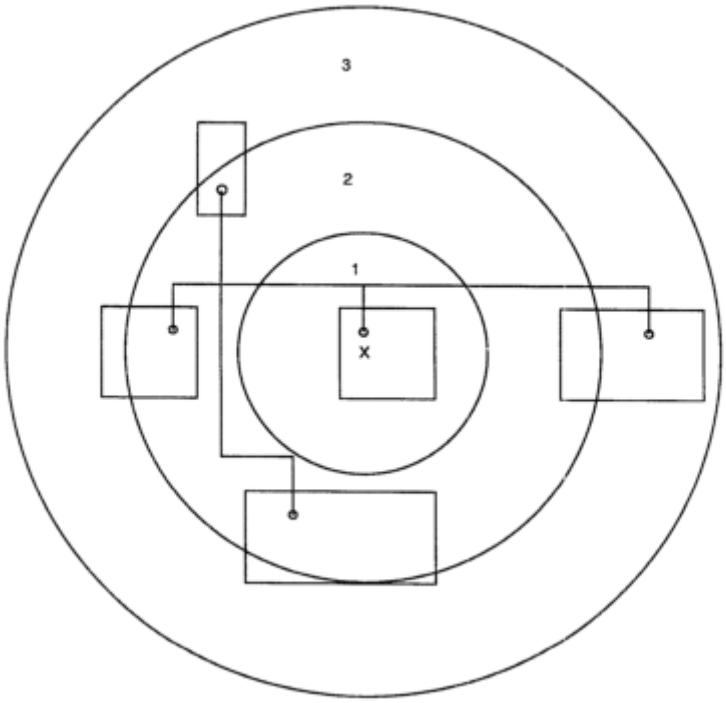
Ανακατανομή. Η ανακατανομή είναι ένα κρίσιμο μέρος του αλγορίθμου. Οι ενότητες που αφαιρούνται πρέπει να ανακατανεμηθούν στις ελεύθερες περιοχές έτσι ώστε να βελτιωθεί η τοποθέτηση. Οι ενότητες που πρέπει να ανακατανεμηθούν ταξινομούνται ανάλογα με τη συνδεσμολογία τους και τοποθετούνται, μια κάθε φορά. Το goodness της κάθε μονάδας σε όλες τις ελεύθερες θέσεις αξιολογείται. Το module τοποθετείται στη θέση που δίνει τη καλύτερη τιμή goodness. Έτσι, οι πιο πυκνά συνδεδεμένες μονάδες θα πάρει την καλύτερη επιλογή για την τοποθεσία κατά τη διάρκεια της ανακατανομής.

Προκαταρκτικά αποτελέσματα από πειραματικά αποτελέσματα δείχνουν ότι ο αλγόριθμος να είναι μια τάξη μεγέθους γρηγορότερος από ό, τι το simulated annealing , με συγκρίσιμη ποιότητα τοποθέτησης.

Εικόνα 33: Η τιμή evaluation of goodness στον αλγόριθμο Kling



Εικόνα 34: Λειτουργία ομόκεντρων κύκλων



4.3 GASP: Genetic Algorithm for Standard Cell Placement

Οι συντάκτες του παρόντος εγγράφου έχουν εφαρμόσει πρόσφατα ένα γενετικό αλγόριθμο για cell placement (GASP) [Shahookar και Mazumder 1990] ως εξής.

4.3.1 Αλγόριθμος

Το διάγραμμα ροής για GASP δίνεται στο Σχήμα 35. Πρώτα, ένας αρχικός πληθυσμός διαμορφώσεων κατασκευάζεται τυχαία. Όλα οι επιμέρους διαμορφώσεις στον πληθυσμό αντιπροσωπεύεται από ένα σύνολο τεσσάρων διανυσμάτων που περιέχουν τον αριθμό των cells , τις x και y-συντεταγμένες, και έναν αύξοντα αριθμό. Οι συντεταγμένες των κελιών προσδιορίζεται με την τοποθέτηση τους τέλος με τέλος στις σειρές. Ο σειριακός αριθμός χρησιμοποιείται για να παρακολουθούμε το κατά προσέγγιση slot στην φυσική περιοχή διάταξης στην οποία κάθε κελί έχει ανατεθεί. Το μέγεθος του πληθυσμού παρέχεται από το χρήστη και καθορίζει την αντίστροφη σχέση μεταξύ του χρόνου επεξεργασίας και της ποιότητας του αποτελέσματος. Από την πειραματική παρατήρηση, διαπιστώθηκε ότι ένας μικρός πληθυσμός των 24 διαμορφώσεων έδωσε την καλύτερη απόδοση. Κάθε άτομο αξιολογείται για να προσδιορίσουμε την καταλληλότητα του. Η καταλληλότητα είναι αμοιβαία και για το μήκους του σύρματος. Ποινές για τον έλεγχο της γραμμής και το μήκος της επικάλυψης των κελιών δεν χρησιμοποιούνται. Αντ 'αυτού, μετά από τη δημιουργία μιας νέας διαμόρφωσης , τα κελιά επαναπροσδιορίζονται για να αφαιρεθεί η

επικάλυψη. Αυτό γίνεται επειδή η αφαίρεση της επικάλυψης δεν παίρνει περισσότερο χρόνο υπολογισμού , σε σχέση με τον προσδιορισμό της ποινής επικάλυψης. Επειδή κατά μέσο όρο τα μισά κελιά μετακινούνται ταυτόχρονα, η πλειοψηφία των δικτύων επηρεάζεται. Επιπλέον, το μήκος σύρματος πρέπει να υπολογίζεται εξαντλητικά, και δεν επιτυγχάνεται εξοικονόμηση επιτρέποντας να έχουμε επικάλυψη.

Στην αρχή της κάθε γενιά, η αναστροφή εκτελείται για κάθε άτομο, με πιθανότητα ίση με το inversion rate. Για το σκοπό αυτό, δύο σημεία τομής καθορίζεται τυχαία, και το τμήμα μεταξύ τους στο πίνακα των cells αναστρέφεται, μαζί με τις συντεταγμένες και τους αύξοντες αριθμούς (σχήμα 31). Στη συνέχεια το crossover λαμβάνει χώρα. Δύο άτομα που επιλέγονται από τον πληθυσμό τυχαία, με μία πιθανότητα ανάλογη της καταλληλότητά τους. Πριν από το crossover, οι σειριακοί αριθμοί του δεύτερου γονέα ευθυγραμμίζονται με την ίδια αλληλουχία όπως εκείνοι του πρώτου γονέα, έτσι τα κελιά στις ίδιες θέσεις του πίνακα αντιστοιχούν σε περίπου τις ίδιες θέσεις επί του τσιπ. Στη συνέχεια, τα τμήματα ανταλλάσσουν μεταξύ τους γονείς, έτσι ώστε για κάθε θέση στο chip, το παιδί να κληρονομεί ένα κελί από τον ένα γονέα ή τον άλλο. Η διαδικασία αυτή επαναλαμβάνεται έως ότου ο επιθυμητός αριθμός απογόνων έχει δημιουργηθεί. Ο αριθμός των απογόνων ανά γενιά, N_s καθορίζεται από το crossover rate

$$N_s = N_p R_c$$

όπου N_p είναι το μέγεθος του πληθυσμού και R_c είναι το ποσοστό crossover. Δεδομένου ότι ο αριθμός των διαμορφώσεων που εξετάστηκαν διατηρείται σταθερός, ο πραγματικός αριθμός των γενεών αυξάνεται καθώς το crossover rate μειώνεται:

$$N_g = \frac{N_{g0} N_p}{N_s},$$

όπου N_p είναι το μέγεθος του πληθυσμού και N_{g0} είναι ο αριθμός των γενεών που καθορίζονται από το χρήστη.

Κάθε απόγονος έχει μεταλλαχθεί με πιθανότητα ίση με το ρυθμό μετάλλαξης. Η Μετάλλαξη αποτελείται από ζεύγη ανταλλαγής. Δύο δύο κελιά επιλέγονται σε τυχαία σειρά και ανταλλάσσουν τις τοποθεσίες που υπάρχουν στους πίνακες, αφήνοντας τα διανύσματα με τις συντεταγμένες αμετάβλητα (Εικόνα 30).

Μετά το crossover και τη μετάλλαξη, η καταλληλότητα του κάθε απογόνου αξιολογείται, και ο πληθυσμός για την επόμενη γενιά επιλέγεται από τον συνδυασμό των γονέων και των απογόνων. Τρεις μέθοδοι επιλογής έχουν δοκιμαστεί: ανταγωνιστική επιλογή, όπου επιλέγονται ο καλύτερος από τους γονείς και ο καλύτερος από τους απογόνους, με τυχαία επιλογή, και με τυχαία επιλογή της διατήρησης του καλύτερου ατόμου.

4.3.2 Crossover Operators

Το crossover είναι η κύρια μέθοδος της βελτιστοποίησης στον γενετικό αλγόριθμο και, στην περίπτωση της τοποθέτησης, λειτουργεί συνδυάζοντας καλά subplacements από δύο διαφορετικές τοποθετήσεις γονέων για να δημιουργήσει μια νέα τοποθέτηση. Για να αντιμετωπίσουμε τις συγκρούσεις που μπορεί να συμβούν στο κλασσικό crossover, πρώτον πρέπει να έναν τρόπο να συνδυάσουμε δύο διαφορετικές τοποθετήσεις χωρίς συγκρούσεις, δεύτερον να χρησιμοποιήσουμε κάποια μέθοδο για την επίλυση των διαφορών που προκύπτουν. Η απόδοση τριών ισχυρών crossover operators έχουν συγκριθεί πειραματικά. Δύο από αυτούς, ο Order και ο PMX, διαφέρουν ως προς την μέθοδο επίλυσης των διαφορών τους, ενώ ο Cycle crossover είναι ένα conflictless operator.

Order Crossover. Ο αλγόριθμος order crossover είναι ως ακολούθως. Επιλέξτε ένα σημείο τομής στην τύχη. Αντιγράψτε πίνακα τμημάτων προς τα αριστερά του σημείου τομής από τον ένα γονέα στους απογόνους. Γεμίστε το υπόλοιπο (δεξί) τμήμα του πίνακα απογόνων με τη μετάβαση μέσω του δεύτερου γονέα, από την αρχή μέχρι το τέλος και τη παίρνοντας τις ενοτήτων που έμειναν εκτός, με τη σειρά. Ένα παράδειγμα παρουσιάζεται στο Σχήμα 36a. Ο εν λόγω operator μεταφέρει ένα subplacement από τον πρώτο γονέα χωρίς καμία αλλαγή, μετά, για την επίλυση των συγκρούσεων, συμπιέζει το δεύτερο γονέα εξαλείφοντας τα κελιά που μεταφέρονται από τον πρώτο γονέα και μετατοπίζοντας τα υπόλοιπα κελιά προς τα αριστερά, χωρίς να αλλάξει η σειρά τους [Davis 1985]. Στη συνέχεια, αντιγράφει το συμπιεσμένο δεύτερο γονέα στο υπόλοιπο μέρος του πίνακα των απογόνων.

PMX. PMX [Goldberg και Lingle 1985] σημαίνει Partial Mapped Crossover(μερικώς χαρτογραφημένο Crossover). Υλοποιείται ως εξής. Επιλέξτε ένα τυχαίο σημείο τομής και να θεωρήστε τα τμήματα που δημιουργήθηκαν μετά την κοπή σε δύο γονείς ως μερική χαρτογράφηση των κελιών που πρέπει να ανταλλάσσονται στον πρώτο γονέα για τη δημιουργία των απογόνων. Πάρτε αντίστοιχα κελιά από τα τμήματα και των δύο γονέων, τοποθετήστε αυτά τα κελιά στο πρώτο γονέα, και κάντε ανταλλαγή. Επαναλάβετε αυτή τη διαδικασία για όλα τα κελιά στο τμήμα. Ετσι, ένα κελί στο τμήμα του πρώτου γονέα και ένα κελί στην ίδια θέση στο δεύτερο γονέα θα καθορίσει πια κελιά του πρώτου γονέα πρέπει να ανταλλαγούν για να δημιουργηθεί οι απόγονοι. Ένα παράδειγμα παρουσιάζεται στο Σχήμα 36b.

Cycle Crossover . Ο cycle crossover [Oliver et al. 1987] είναι μια προσπάθεια για να εξαλείψουμε τις συγκρούσεις των κελιών που συνήθως προκύπτουν σε crossover operators. Στους απογόνους που παράγονται από cycle crossover , κάθε κελί είναι στην ίδια θέση όπως ήταν στον ένα γονέα ή στον άλλο. Για cycle crossover , ξεκινάμε από το κελί στη θέση 1 του γονέα 1 (ή οποιοδήποτε άλλο σημείο αναφοράς) και να το αντιγράψουμε στη θέση 1 των απογόνων. Τώρα σκεφτείτε τι θα συμβεί στο κελί στη θέση 1 του γονέα 2. Ο απόγονος δεν μπορεί να κληρονομήσει αυτό το κελί από γονέα 2, εφόσον η θέση 1 στους απογόνους έχει γεμίσει. Έτσι, αυτό το κελί θα πρέπει να αναζητηθεί στο γονέα 1 και να περάσει στους απογόνους από εκεί. Υποθέτοντας ότι αυτό το κελί βρίσκεται στο γονέα 1 στη τοποθεσία x . Τότε, περνά στους απογόνους στη τοποθεσία χ . Αλλά τότε το κελί στη θέση χ στο γονέα 2 δεν μπορεί να περάσει στους απογόνους, έτσι το κελί περνά επίσης από το γονέα 1. Αυτή η διαδικασία συνεχίζεται έως ότου έχουμε ολοκληρώσει έναν κύκλο και

φτάσουμε σε ένα κελί που έχει ήδη περάσει. Στη συνέχεια, επιλέγουμε ένα κελί από το γονέα 2 να περάσει στους απογόνους και περνάμε από έναν ακόμη κύκλο, περνώντας τα κελιά από τον γονέα 2 στους απογόνους. Έτσι, σε εναλλασσόμενους κύκλους, ο απόγονος κληρονομεί κελιά από εναλλακτικούς γονείς, και τα κελιά τοποθετούνται στις ίδιες θέσεις όπως ήταν στους γονείς από τους οποίους είχαν κληρονομήσει. Ένα παράδειγμα δίδεται στο Σχήμα 36c.

4.3.3 Πειραματικά αποτελέσματα

Στις περισσότερες περιπτώσεις, είτε ο PMX είτε ο cycle crossover είχαν τις καλύτερες επιδόσεις, ενώ ο order crossover ήταν ο χειρότερος. Ο cycle crossover βρέθηκε να είναι ελαφρώς καλύτερος από ό, τι ο PMX. Ο καλύτερος συμβιβασμός των παραμέτρων ήταν, crossover rate 33%, inversion rate 15% και mutation rate 0.5% . Αυτές οι τιμές χρησιμοποιήθηκαν σε όλα τα επόμενα πειράματα. Σε όλες τις περιπτώσεις, η competitive selection των καλυτέρων γονέων και απογόνων να συμπεριληφθούν στην επόμενη γενεά αποδείχθηκε ότι είναι καλύτερη από ό, τι όλες οι άλλες στρατηγικές. Στα σχήματα 37a-c έχουμε τις γραφικές παραστάσεις που δείχνουν τα χαμηλότερα και μεσαία κόστη καλωδίωσης σε κάθε γενιά καθώς η βελτιστοποίηση προχωρά. Ο λόγος για την κακή απόδοση των μεθόδων τυχαίας επιλογής μπορεί να φανεί εμφανώς . Ακριβώς όπως είναι δυνατόν να συνδυαστούν τα καλά χαρακτηριστικά των δύο γονέων για να σχηματίσουμε ένα καλύτερο απόγονο , είναι επίσης δυνατόν να συνδυάσουμε τα κακά χαρακτηριστικά των γονέων και να δημιουργήσουμε έναν πολύ κακό απόγονο. Αν αυτοί οι απόγονοι γίνουν δεκτοί σε μια τυχαία βάση, στο καλύτερο κόστος αλλά και στο μέσο κόστος στον πληθυσμό θα είχαμε αυξομειώσεις, όπως φαίνεται στο Σχήμα 37c. Οι απώλειες που έχουμε στην τυχαία διαδικασία υπερβαίνουν κατά πολύ οποιοδήποτε πλεονέκτημα έχει αποκτηθεί, και ο αλγόριθμος παίρνει πολύ περισσότερο χρόνο για να συγκλίνει. Όταν επιτρέπουμε την διατήρηση της βέλτιστης λύσης μαζί με τυχαία επιλογή, το κόστος της βέλτιστης λύσης φαίνεται να μειώνεται μονοτονικά. Το μέσο κόστος του πληθυσμού αυξομειώνεται ακόμη, ωστόσο, και η σύγκλιση είναι πολύ βραδύτερη σε σχέση με την ανταγωνιστική επιλογή.

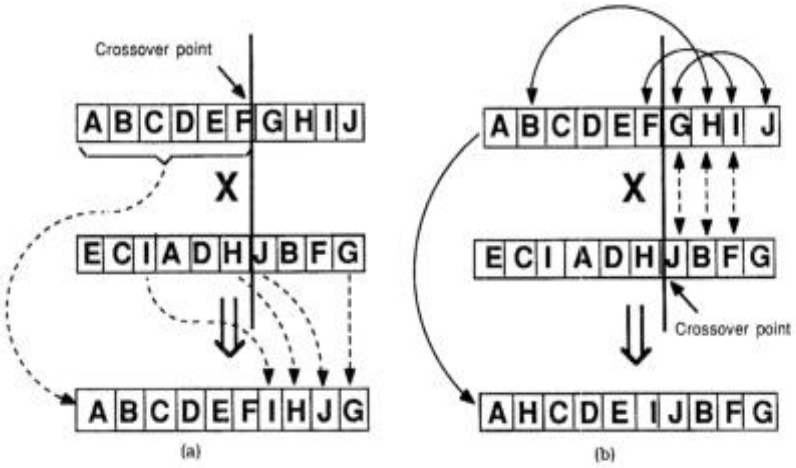
Comparison with TimberWolf. Η απόδοση του αλγορίθμου συγκρίθηκε με τον 3.3 Timberwolf για πέντε κυκλώματα που κυμαίνονται από 100 έως 800 κελιά. Πέτυχε την ίδια ποιότητα της τοποθέτησης σε περίπου ίδιο χρόνο CPU. Υπάρχουν δύο ενδιαφέρουσες διαφορές, εντούτοις.

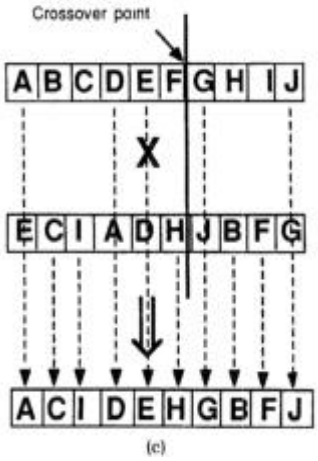
Ο GASP πέτυχε μια πολύ γρήγορη βελτίωση στην αρχή, στη συνέχεια, όμως έπεσε επίπεδο, όπως απεικονίζεται στο Σχήμα 38. Από την άλλη πλευρά, για τον Timberwolf το κόστος αυξάνεται στις πρώτες επαναλήψεις με τις υψηλές θερμοκρασίες, και μικρή βελτίωση επιτυγχάνεται κατά το πρώτο μισό του τρεξίματος. Αυτό σημαίνει ότι εάν δεν χρειάζεται μια πολύ υψηλής ποιότητας τοποθέτηση δεν είναι απαραίτητο, ο GASP θα είναι πολλές φορές ταχύτερος.

Μια άλλη διαφορά είναι ότι , αν και οι χρόνοι CPU ήταν συγκρίσιμοι , ο GASP εξέτασε 20 - 50 φορές λιγότερες διαμορφώσεις για την ίδια ποιότητα της τοποθέτησης. Αυτό το πλεονέκτημα αντισταθμίζεται από το υπερβολικά μεγάλο χρονικό διάστημα αξιολόγησης, το οποίο είναι η συμφόρησης του αλγορίθμου. Σε simulated annealing, κινούνται μόνο δύο κελιά κάθε φορά, έτσι ώστε μόνο μερικά

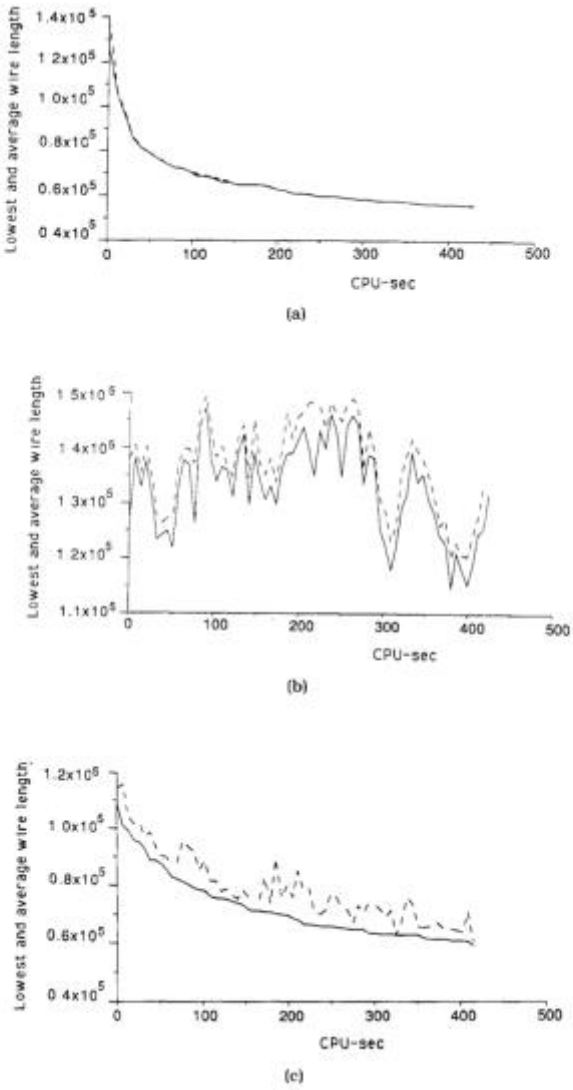
neta πρέπει να αξιολογούνται για να προσδιοριστεί η μεταβολή στο μήκος του καλωδίου. Στο GASP, σχεδόν το ήμισυ των κελιών μετακινούνται ταυτόχρονα, και το μήκος καλωδίου πρέπει να υπολογισθεί εξαντλητικά. Αυτό παίρνει 62-67% του χρόνου της CPU, ενώ το crossover παίρνει μόνο το 17% του χρόνου.

Εικόνα 36: Crossover operators σε GASP.

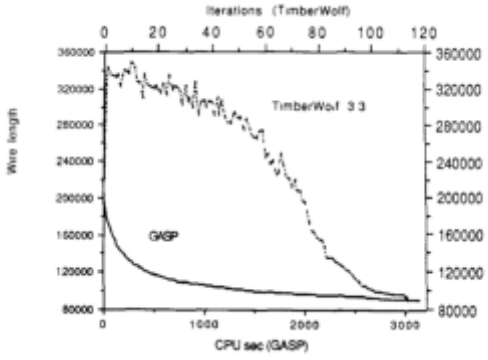




Εικόνα 37: Σύγκριση μεθόδων GASP.



Εικόνα 38: Βελτιστοποίηση GASP σε σχέση με τον TimberWolf



5. Συμπεράσματα

Στο έγγραφο αυτό συζητήθηκαν πέντε κατηγορίες αλγορίθμων VLSI τοποθέτησης module . Ο simulated annealing είναι σήμερα ο πιο δημοφιλής μεταξύ των ερευνητών και είναι ο καλύτερη διαθέσιμος αλγόριθμος από την άποψη της ποιότητας τοποθέτησης, αλλά χρειάζεται υπερβολικό χρόνο υπολογισμού. Συζητήσαμε τον αλγόριθμο Timberwolf 3.2 από τον Sechen, βελτιώσεις που έγιναν στο Timberwolf 4.2 , και άλλες πρόσφατες εξελίξεις, όπως των πειραμάτων σχετικά με το πρόγραμμα ψύξης από τον Nahar et al [1985] και την ανάλυση της αλυσίδας Markov από τον Aarts et al.

Οι Min-cut αλγόριθμοι κατέχουν τη δεύτερη θέση όσον αφορά την ποιότητα τοποθέτησης αλλά θα μπορούσαν πιθανότατα είναι οι καλύτεροι από την άποψη της σχέσης κόστους / απόδοσης, δεδομένου ότι είναι πολύ ταχύτεροι από ό, τι ο simulated annealing . Αυτοί οι αλγόριθμοι βασίζονται σε μια απλή αρχή, οι ομάδες των κελιών που είναι πυκνά συνδεδεμένες μεταξύ τους θα πρέπει να τοποθετηθούν κοντά μαζί. Έτσι, από επανειλημμένες διαμερίσεις του συγκεκριμένου δικτύου για την ελαχιστοποίηση του net-cut και κάθε φορά που περιορίζει τις υποομάδες που θα τοποθετηθούν σε διάφορες περιοχές της διάταξης, το μήκος του καλωδίου είναι ελαχιστοποιημένο. Οι αλγόριθμοι του Breuer έχουν συζητηθεί σε αυτό το έγγραφο, μαζί με βελτιώσεις όπως διάδοση τερματικού από τον Dunlop και Kernighan [1985], και quadrisection από τον Suaris και Kedem [1987].

Οι Force-directed αλγόριθμοι λειτουργούν με τη φυσική αναλογία των μαζών που συνδέονται με springs,όπου το σύστημα θα τείνει να ακινητοποιηθεί στην ελάχιστη κατάσταση ενέργειας, με ελάχιστη τάση στα springs, είτε σε όρους προβλήματος τοποθέτησης, το μήκος του καλωδίου να ελαχιστοποιείται. Οι Force-directed αλγόριθμοι υπάρχουν από τη δεκαετία του 1960 και ήταν από τους πρώτους αλγορίθμους που χρησιμοποιήθηκαν για την τοποθέτηση Μια πλούσια ποικιλία από εφαρμογές έχουν αναπτυχθεί τα τελευταία χρόνια, συμπεριλαμβανομένης της δομικής (εξίσωση επίλυσης) μεθόδου για τον προσδιορισμό μιας ελάχιστης ενέργειας διαμόρφωσης από το μηδέν και δύο τύπους επαναληπτικών τεχνικών, ο ένας αποτελείται από την επιλογή ενοτήτων, ένα κάθε φορά και για τον καθορισμό μιας ιδανικής τοποθεσίας για αυτούς από το force consideration και η άλλη που αποτελείται από τυχαία / εξαντλητική ανταλλαγή κατά ζεύγη, με αποδοχή των καλών κινήσεων και απόρριψη των κακών κινήσεων, και πάλι με βάση το force consideration. Μια επισκόπηση των διάφορων τεχνικών που χρησιμοποιούνται έχει δοθεί, μαζί με ένα δείγμα αλγόριθμου και ένα παράδειγμα δικτύου να απεικονίζει τη λειτουργία του αλγορίθμου. Ο Goto’s GFDR αλγόριθμος έχει επίσης συζητηθεί .

Η τοποθέτηση είναι ένα πρόβλημα βελτιστοποίησης, και μέθοδοι όπως ο Simplex, ο Quadratic Programming, και η Penalty Function μέθοδος έχουν παραδοσιακά χρησιμοποιηθεί για διάφορα γραμμικά και μη γραμμικά προβλημάτα βελτιστοποίησης. Περαιτέρω, το πρόβλημα της τοποθέτησης μπορεί επίσης να διαμορφωθεί με όρους του προβλήματος της τετραγωνικής εκχώρησης, το οποίο μπορεί να λυθεί με τη μέθοδο των ιδιοτιμών. Συνεπώς, οι διάφορες εργασίες που χρησιμοποιούν αυτές τις τεχνικές έχουν συζητηθεί κάτω από την κατηγορία των αριθμητικών τεχνικών βελτιστοποίησης. Το κοινό χαρακτηριστικό όλων αυτών των τεχνικών είναι ότι δεν περιορίζουν τις ενότητες στα σημεία του δικτύου ή των

γραμμών, ως εκ τούτου, είναι πιο εφαρμόσιμα για macroblocks από ό, τι με τα standard cells ή πίνακες πυλών, αν και η λύση που δημιουργείται με αριθμητικές τεχνικές μπορεί να επεξεργαστεί περαιτέρω και να χαρτογραφήσει τις ενότητες στα πιο κοντινά grid points.

Η τελική κλάση των αλγορίθμων που συζητήθηκαν εδώ είναι οι γενετικοί αλγόριθμοι, που, αν και εφευρέθηκε στη δεκαετία του 1960, δεν είχαν χρησιμοποιηθεί για τοποθέτηση μέχρι το 1986. Ο γενετικός αλγόριθμος έχει μια εξαιρετικά αποδοτική αναζήτηση και τεχνική βελτιστοποίησης για προβλήματα με ένα μεγάλο και ποικίλο χώρο αναζήτηση, καθώς και τα προβλήματα όπου περισσότερα από ένα φυσικά χαρακτηριστικά χρειάζεται να βελτιστοποιηθούν ταυτόχρονα. Ο γενετικός αλγόριθμος επεξεργάζεται ένα σύνολο εναλλακτικών τοποθετήσεων μαζί και δημιουργεί μια νέα τοποθέτηση για δοκιμή, συνδυάζοντας subplacements από δύο τοποθετήσεις γονέων. Αυτό προκαλεί τη κληρονομικότητα και τη συσσώρευση των καλών subplacements από τη μια γενιά στην επόμενη. Προκαλεί επίσης την ανάμιξη των καλών χαρακτηριστικών αρκετών διαφορετικών τοποθετήσεων που γίνονται ταυτόχρονα με αποτέλεσμα το αμοιβαίο όφελος. Έτσι, η αναζήτηση μέσα από το χώρο λύσεων είναι εγγενώς παράλληλο. Το πρόβλημα της τοποθέτησης παριστάνεται με τη μορφή ενός γενετικού κώδικα, καθώς και οι γενετικοί operators λειτουργούν με αυτόν τον κώδικα, δεν συνδέονται άμεσα με τη φυσική διάταξη. Αυτό είναι μια σημαντική απόκλιση από τους συμβατικούς αλγορίθμους τοποθέτησης που ισχύουν άμεσα οι μετασχηματισμοί με την φυσική διάταξη. Αυτός ο εγγενής παραλληλισμός του γενετικού αλγορίθμου μπορεί, ωστόσο, να είναι ένα δυνητικό πρόβλημα, εκτός αν ένα έξυπνο σύστημα αναπαράστασης έχει σχεδιαστεί για να αντιπροσωπεύει τη φυσική τοποθέτηση ως γενετικό κώδικα, ο αλγόριθμος μπορεί να αποδειχθεί αναποτελεσματικός. Στο έγγραφο αυτό, τρεις εφαρμογές του γενετικού αλγορίθμου που ξεπερνούν αυτά προβλήματα με διαφορετικούς τρόπους έχουν περιγραφεί.

Ο Πίνακας 5 είναι μία προσεγγιστική σύγκριση της απόδοσης των αλγορίθμων που συζητούνται εδώ. Ο πίνακας 6 δίνει το χρόνο εκτέλεσης και την απόδοση ορισμένων από τους αλγορίθμους. Το μήκος του καλωδίου ή περιοχή του τσιπ στη στήλη των επιδόσεων είναι κανονικοποιημένο. Αυτά τα δεδομένα μπορούν να δώσουν μόνον μερικές συγκρίσεις, αφού διαφορετικά papers έχουν δώσει αποτελέσματα σε διαφορετικά κυκλώματα και έχει χρησιμοποιηθεί διαφορετικό υλικό υπολογιστή. Μια προσπάθεια έχει γίνει να ομαδοποιήσουμε τα στοιχεία σύμφωνα με το υλικό του υπολογιστή που χρησιμοποιείται.

Πίνακας 5. Σύγκριση αλγόριθμων τοποθέτησης

Algorithm	Result quality	Speed
Simulated annealing	Near optimal	Very slow
Genetic algorithm	Near optimal	Very slow
Force directed	Medium . . . good	Slow . . . medium
Numerical optimization	Medium . . . good	Slow . . . medium
Min-cut	Good	Medium
Clustering and other constructive placement	Poor	Fast

Πίνακας 5. Σύγκριση χρόνου εκτέλεσης αλγόριθμων τοποθέτησης

Implementation	algorithm	No. of cells	CPU hours	Computer hardware	Performance	Reference
Huang et al.	Simulated	469	1.42	VAX 11/780	Wire lengths	[Huang et al. 1986]
TimberWolf 3.2	Annealing	469	3		within ±4%	
Huang		800	10.42			
TimberWolf 3.2		800	10.7			
Dunlop and Kernighan	Min-cut	412	1	VAX 11/780	Comparable to manual layout	[Dunlop and Kernighan 1985]
Quadrisection	Min-cut	173	0.01	VAX 8600	Chip area = 1.11	[Suaris and Kedem 1987]
TimberWolf 3.2		173	0.53		Chip area = 1.0	
Quadrisection		796	0.135		Chip area = 0.91	
TimberWolf 3.2		796	17.8		Chip area = 1.0	
Proud-2	Gauss-	1438	0.014	VAX 8650	Wire length = 0.93	[Tsay et al. 1988]
Proud-4	Seidel	1438	0.027		Wire length = 0.9	
TimberWolf 3.2		1438	2		Wire length = 1.0	
TimberWolf 4.2		1438	0.9		Wire length = 0.84	
Proud-2		3816	0.09		Wire length = 0.90	
Proud-4		3816	0.18		Wire length = 0.91	
TimberWolf 3.2		3816	—		Wire length = 1.0	
TimberWolf 4.2		3816	6.69		Wire length = 0.83	
Proud-2		26277	0.85		Wire length = 1.0	
Proud-4		26277	1.56		Wire length = 0.962	
ESP	Evolution	183	0.43	Sun 3/75	Wire length = 1.0	[Kling 1987]
TimberWolf 3.2		183	2.7		Wire length = 1.0	
GASP	Genetic	469	11.0	Apollo-	Wire length = 1.0	[Shahookar and
TimberWolf 3.2		469	11.3	DN4000	Wire length = 1.02	Mazumder 1990]
GASP		800	12.5		Wire length = 1.0	
TimberWolf 3.2		800	13.7		Wire length = 0.87	

Παρά την απίστευτη ποικιλία διαθέσιμων αλγορίθμων, αποδοτική τοποθέτηση module παραμένει μέχρι στιγμής ένας απατηλός στόχος. Τα περισσότερα από τα ευρετικά που έχουν δοκιμαστεί καταναλώνουν υπερβολικές ποσότητες χρόνου CPU και παράγουν μη βέλτιστα προϊόντα. Μέχρι πρόσφατα οι υπερβολικοί χρόνοι υπολογισμού είχαν απαγορεύσει την επεξεργασία των κυκλωμάτων με περισσότερες από μερικές χιλιάδες μονάδες. Τα πρώτα αποτελέσματα, δείχνουν ότι αυτοί οι αλγόριθμοι έχουν την ικανότητα να παράγουν σχεδόν βέλτιστες τοποθετήσεις σε εύλογο χρονικό διάστημα υπολογισμού.

Το παρακάτω είναι μια λίστα από άλλες έρευνες και σεμινάρια για την τοποθέτηση cells σε χρονολογική σειρά: Hanan και Kurtzberg [1972], Press [1979], Soukup [1981], Chew [1984], Hu και Kuh [1985], Hildebrandt [1985], Goto και Matsuda [1986], και Press Karger [1986], Sangiovanni-Vincentelli [1987], Wongetal [1988], και Press και Lorenzetti [1988].

Robson [1984] και VLSI [1987, 1988] λίστα διεξοδικών ερευνών για εμπορικά διαθέσιμο λογισμικό αυτόματης διάταξης. Οι έρευνες δείχνουν ότι η force-directed τοποθέτησης ήταν ο αλγόριθμος που επιλέγαμε σε συστήματα διαθέσιμα το 1984 [Robson 1984]. Το 1987 και το 1988, βλέπουμε ένα μίγμα του force-directed

αλγορίθμου, του min-cut, καθώς και του simulated annealing[VLSI 1987, 1988]. Σύμφωνα με την έρευνα του 1988, μερικά από αυτά τα συστήματα μπορούν να χρησιμοποιηθούν για να τοποθετήσουν και να δρομολογήσουν μια θάλασσα από πίνακες πυλών με περισσότερες συστοιχίες από 100.000 πύλες, σε τριπλό μέταλλο, χρησιμοποιώντας μέχρι το 80% των διαθέσιμων πυλών [VLSI 1988]. Μια άλλη τάση εμφανή από αυτές τις έρευνες είναι ότι σχεδόν όλα τα συστήματα μπορούν να τρέξουν σε επιτραπέζιους σταθμούς εργασίας-Sun, Apollo, ή Microvax. Έτσι, τα αυτοματοποιημένα συστήματα διάταξης είναι ευρέως διαθέσιμα.

Αναφορές

AARTS, E. H. L , DEBONT, F. M. J., AND HABERS, E. H. A. 1985. Statistical cooling: A general approach to combinatorial optimization problems. PhilLps J. Res. 40, 4, 193-226.

AARTS, E. H. L., DEBONT, F. M. J., AND HABERS, E. H. A. 1986. Parallel implementations of the statistical cooling algorithm. Integration,VLSI J. 4, 3 (Sept.) 209-238.

AKERS, S. B. 1981. On the use of the linear assignment algorithm in module placement. In Proceedings of the 18th Design Automation Conference. pp. 137-144.

ANTREICH, K. J., JOHANNES, F. M., AND KIRSCH, F H. 1982. A new approach for solving the placement problem using force models. In Proceedings of the IEEE International Symposmm on Circuits and Systems. pp. 481-486.

BANNERJEE, P., AND JONES, M. 1986. A parallel simulated annealing algorithm for standard cell placement on a hypercube computer. In Proceedings of the IEEE International Conference on Computer Design. p. 34.

BENDERS, J. F. 1962. Partitioning procedures for solving mixed variable problems. Numer, Math. 4, 238-252.

BLANKS, J. P. 1984. Initial placement of gate arrays using least squares methods. In Proceedings of the 21st Design Automation Conference. pp. 670-671.

BLANKS, J, P. 1985a. Near-optimal placement using a quadratic objective function. In Proceedings of the 22nd Design Automation Conference. pp. 609-615.

BLANKS, J. P. 19S5b. Use of a quadratic objective function for the placement problem in VLSI design. Ph.D. dissertation, Univ. of Texas at Austin.

BREUER, M. A. 1977a. Min-cut placement, J. Design Automation and Fault- Tolerant Computing 1, 4 (Oct.) 343-382.

BREUER,M. A. 1977b. A class of min-cut placement algorithms. In Proceedings of the 14th Design Automation Conference. pp. 284-290

CASSOTO,A., ROMEO, F., AND SANGIOVANNI - VINCENTELLI,A. 1987. A parallel simulated annealing algorithm for the placement of standard cells. IEEE Trans. Comput.-Aided Design CAD-6, 5 (May), 838.

CHAN, H. M., AND MAZUMDER, P. 1989. A genetic algorithm for macro cell placement. Tech. Rep.
Computing Research Laboratory, Dept. of Electrical Engineering and Computer Science, University
of Michigan, Ann Arbor, Mich.

CHANG, S. 1972. The generation of minimal trees with a steiner topology. J. ACM 19, 4 (Oct.), 699-711.

CHEN, N. P. 1983. New algorithms for steiner tree on graphs. In Proceedings of the International
Symposium on Circuits and Systems. pp. 1217-1219.

CHENG, C. 1984. Placement algorithms and applications to VLSI design. Ph.D. dissertation Dept. of Electrical Engineering, Univ. of California, Berkeley.

CHENG, C., AND KUH, E. 1984. Module placement based on resistive network optimization. IEEE Trans. Comput.-Aided Design CAD-3, 7 (July), 218-225.

CHUNG, M. J., AND RAO, K. K. 1986. Parallel simulated annealing for partitioning and routing. In Proceedings of the IEEE International Conference on Computer Design. pp. 238-242.

CHYAN, D., AND 13REUER,M. A. 1983. A placement algorithm for array processors. In Proceedings of the 20th Design Automation Conference. pp. 182-188.

COHOON, J. P., AND SAHNI, S. 1983. Heuristics for the board permutation problem. In Proceedings of the 20th Design Automation Conference.

COHOON, J. P., AND PARIS, W. D. 1986. Genetic placement. In Proceedings of the IEEE International Conference on Computer-Aided Design. pp. 422-425.

CORRIGAN, L. I. 1979. A placement capability based on partitioning. In Proceedings of the 16th Design Automation Conference. pp. 406-413.

DAVIS, L. 1985. Applying adaptive algorithms to epistatic domains. In Proceedings of the International Joint Conference on Artificial Intelligence.

DONATH, W. E. 1980. Complexity theory and design automation. In Proceedings of the 17th Design Automation Conference. pp. 412-419.

DUNLOP, A. E., AND KERNIGHAN, B. W. 1985. A procedure for placement of standard cell VLSI circuits. IEEE Trans. Comput. -Aided Design CAD-4, 1 (Jan.), 92-98.

FIDUCCIA, C. M., AND MATTHEYSES, R. M. 1982. A linear-time heuristic for improving network partitions. In Proceedings of the 19th Design Automation Conference. pp. 175-181.

FISK, C. J., CASKEY, D. L., AND WEST, L. E. 1967. Accel: Automated circuit card etching layout. Proc. IEEE 55, 11 (Nov.) 1971-1982.

FUKUNAGA, K., YAMADA, S., STONE, H., AND KASAI,T. 1983. Placement of circuit modules using a graph space approach. In Proceedings of the 20th Design Automation Conference. 465-473.

GIDAS, B. 1985. Non-stationary Markov chains and convergence of the annealing algorithm. J. Stat. Phys. 39, 73-131.

GILMORE, P. C. 1962. Optimum and suboptimum algorithms for the quadratic assignment problem. J. SIAM 10, 2 (June), 305-313.

GOLDBERG, D. E., AND LINGLE, R. 1985. Alleles,loci and the traveling salesman problem. In Proceedings of the International Conference on Genetic Algorithms and their Applications.

GOTO, S. 1981. An efficient algorithm for the two-dimensional placement problem in electrical circuit layout. IEEE Trans. Circuits Syst., CAS-28 (Jan.), 12-18.

GOTO, S., AND KUH, E. S. 1976. An approach to the two-dimensional placement problem in circuit layout. IEEE Trans. Circuits Syst. CAS-25, 4, 208-214.

GOTO, S., CEDERBAUM, I., AND TING, B.S. 1977. Suboptimal solution of the backboard ordering with channel capacity constraint. IEEE Trans. Circuits Syst. (Nov. 1977), 645-652.

GOTO, S., AND MATSUDA, T. 1986. Partitioning, assignment and placement. In Layout Design And Verification, T. Ohtsuki, Ed. Elsevier North-Holland, New York, Chap. 2, pp. 55-97.

GREENE, J. W., AND SUPOWIT, K. J. 1984. Simulated annealing without rejected moves. In Proceedings of the IEEE International Conference on Computer Design. pp. 658-663.

GREFENSTETTJE., J., Ed. 1985. In Proceedings of an International Conference on Genetic Algorithms and their Applications. Pittsburgh,Penn.

GREFENSTETTJE., J., Ed. 1987. In Proceedings of the 2nd International Conference on Genetic Algorithms and their Applications. Cambridge, Mass.

GROVER, L. K. 1987. Standard cell placement using simulated sintering, In Proceedings of the 24th Design Automation Conference. pp. 56-59.

HAJEK, B. 1988. Cooling schedules for optimal annealing. Oper. Res. 13, 2 (May), 311-329.

HALL, K. M. 1970. An r-dimensional quadratic placement algorithm. Manage. Sci. 17, 3 (Nov.), 219-229.

HANAN, M., AND KURTZBERG, J. M. 1972a. Placement techniques. In Design Automation of Digital Systems, 1, M A. Breuer, Ed. Prentice Hall, Englewood Cliffs, N. J., Chap. 5, pp. 213-282.

HANAN, M., AND KURTZBERG, J. M. 1972b. A review of placement and quadratic assignment problems. SIAM Reu. 14, 2 (Apr.), 324-342.

HANAN, M., AND WOLFF, P. K., AND AGULE, B. J. 1976a. Some experimental results on placement techniques. In Proceedings of the 13th Design Automation Conference. pp. 214-224.

HANAN, M., AND WOLFF, P. K., AND AGULE, B. J. 1976b. A study of placement techniques. J. Design Automation and Fault-Tolerant Computing 1, 1 (Oct.), 28-61.

HANAN, M., WOLFF, P. IK., AND AGULE, B. J. 1978. Some experimental results on placement techniques. J. Design Automation and Fault-Tolerant Computing 2 (May), 145-168.

HERRIGEL, A., AND FICHTNER, W. 1989. An analytic optimization technique for placement of macrocells. In Proceedings of the 26th Design Automation Conference. pp. 376-381.

HILDEBRANDT, T. 1985. An annotated placement bibliography. ACM SIGDA Newsletter 15, 4 (Dec.), 12-21.

HILLNER, H., WEIS, B. X., AND MLYNSKI, D. A. 1986. The discrete placement problem: A dynamic
programming approach. In Proceedings of the International Symposuim on Circuits and Systems. pp. 315-318.

HOLLAND, J. H. 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, Mich.

HU, T. C., AND KUH, E. S. 1985. VLSI Circuit Layout. IEEE Press, New York.

HUANG, M. D., ROMEO, F., AND SANGIOVANNI - VINCENTELLI, A. 1986. An efficient general cooling schedule for simulated annealing. In Proceedings of the IEEE International Conference
on Computer-Aided Design. pp. 381–384.

HWANG, F. K 1976. “On Steiner Minimal Trees with Rectilinear Distance,” SIAM J. Appl. Math. Vol. 30, PP.104-114, 1976

HWANG, F. K. 1979. An O(n log n) algorithm for suboptimal rectilinear steiner trees. IEEE Trans. Circuits Syst. CAS-26, 1, 75-77.

JARMON, D. 1987. An automatic placer for arbitrary sized rectangular blocks based on a cellular model. In Proceedings of the IEEE International Conference on Computers and Applications. pp. 842-845.

JOHANNES, F. M., JUST, K. M., AND ANTREICH, K. J. 1983. On the force placement of logic arrays. In Proceedings of the 6th European Conference on Circuit Theory and Design. pp. 203-206.

JOHNSON, D. B., AND MIZOGUCHI, T. 1978. Selecting the *k*th element in X + Y and $X_1 + X_2 + \ldots + X_m$. SIAM J. Comput. 7, 2 (May), 141-143

KAMBE, T., CHIBA, T., KIMURA, S., INUFUSHI, T ,OKUDA, N., AND NISHIOKA, I. 1982. A placement algorithm for polycell LSI and its evaluation. In Proceedings of the 19th Design Automation Conference. PP 655-662

KANG, S. 1983. Linear ordering and application to placement. In Proceedings of the 20th Design Automation Conference. pp. 457-464.

KAPPEN, H. J., AND DE BONT, F. M. J. 1990. An efficient placement method for large standard-cell
and sea-of-gates designs. In Proceedings of the IEEE European Design Automation Conference. pp. 312-316.

KARGER, P. G., AND MALEK, M. 1984. Formulation of component placement as a constrained optimization problem. In Proceedings of the International Conference on Computer Design. pp. 814-819.

KERNIGHAN,B. W., ANDLIN, S. 1970. An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. 49, 2, 291-308.

KIRKPATRICK, S., GELATT, C D , AND VECCHI, M P. 1983. Optimization by simulated annealing. Science 220.4598 (May), 671-680.

KLING, R. M, 1987. Placement by simulated evolution. Master's thesis, Coordinated Science Lab, College of Engr., Univ. of Illinois at Urbana-Champaign.

KLING, R., AND BANNERJEE, P. 1987. ESP: A new standard cell placement package using simulated evolution. In Proceedings of the 24th Design Automation Conference. pp. 60-66.

KOZAWA, T., MIURA, T., AND TERAI, H. 1984. Combine and top down block placement algorithm for hierarchical logic VLSI layout. In Proceedings of the 21st Design Automation Conference. pp. 535-542.

KOZAWA, T , TERAI, H., ISHII, T., HAYASE, M., MIURA,C., OGAWA, Y , KISHIDA, K., YAMADA, N., AND OHNO, Y. 1983. Automatic placement algorithms for high packing density VLSI. In Proceedings of the 20th Design Automation Conference. pp. 175-181

KRUSKAL, J. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. In proceedings of the American Mathematical Society, Vol. 7, No. 1, pp. 48-50.

VAN LAARHOVEN, P. J. M., AND AARTS, E. H. L. 1987. Simulated Annealing: Theory and Applications. D. Riedel, Dordrecht-Holland.

LAM, J., AND DELOSME, J. 1986. Logic minimization using simulated annealing. In Proceedings of the IEEE International Conference on Computer-Aided Design. p. 378.

LAM, J., AND DELOSME, J. 1988. Performance of a new annealing schedule. In Proceedings of the 25th Design Automation Conference. pp.306-311.

LAUTHER, U. 1979. A min-cut placement algorithm for general cell assemblies based on a graph representation. In Proceedings of the 16th Design Automation Conference. pp. 1-10.

LEIGHTON, F. T. 1983. Complexity Issues in VLSI. MIT Press, Cambridge, Mass.

LUNDY, M., AND MEES, A. 1984 Convergence of the annealing algorithm. In proceedings of the Simulated Annealing Workshop.

MAGNUSON, W. G. 1977. A comparison of constructive placement algorithms. IEEE Region 6 Conf, Rec. 28-32.

MALLELA, S., AND GROVER, L. K. 1988. Clustering based simulated annealing for standard cell
placement. In Proceedings of the 25th Design Automation Conference. pp. 312-317.

MARKOV, L. A., Fox, J. R., AND BLANK, J. H. 1984. Optimization technique for two-dimensional placement. In Proceedings of the 21st Design Automation Conference. pp. 652-654.

MITRA, D., RONIEO, F., AND SANGIOVANNI-VINCENTELLI, A. 1985. Convergence and finite-time behavior of simulated annealing. In Proceedings of the 24th Conference on Decision and Control. pp. 761-767.

MOGAKI, M., MWRA, C., AND TERAI, H. 1987. Algorithm for block placement with size optimization technique by the linear programming approach. In Proceedings IEEE International Conference on Computer-Aided Design. pp. 80-83.

MUROGA, S. 1982. VLSI System Design. John Wiley, New York, Chap. 9, pp. 365-395.

NAHAR, S., SAHNI, S., AND SHRAGOWITZ, E. 1985, Experiments with simulated annealing. In Proceedings of the 22th Design Automation Conference. pp. 748-752.

OLIVER, I. M., SMITH, D. J., AND HOLLAND, J. R. C. 1985. A study of permutation crossover operators on the traveling salesman problem. In Proceedings of the International Conference on Genetic Algorithms and their Applications. pp. 224-230.

OTTEN, R., AND VAN GINNEKIN,L. 1984. Floorplan design using simulated annealing. In Proceedings of the IEEE International Conference on Computer-Aided Design. pp. 96-98

PALCZEWSKI, 1984. Performance of algorithms for initial placement. In Proceedings of the 21st
Design Automation Conference, pp. 399-404.

PERSKY, G., DEUTSCH, D. N., AND SCHWEIKERT,D. J., 1976. LTX: A system for the directed automatic design of LSI circuits. In Proceedings of the 13th Design Automation Conference. pp. 399-407.

PREAS, B. T. 1979. Placement and routing algorithms for hierarchical integrated circuit layout Ph.D. dissertation, Dept. of Electrical Engr., Stanford Univ. Also Tech. Rep. 180, Computer Systems Lab, Stanford Univ.

PREAS, B. T., AND KARGER, P. G. 1986. Automatic placement: A review of current techniques. In
Proceedings of the 23rd Design Automation Conference. pp. 622-629.

PREAS, B., AND LORENZETTI, M. 1988. Placement, assignment and floorplanning. In 20Physical
Design Automation of VLSI Systems. The Benjamin Cummings Publishing Co., Menlo Park, Calif., Chap. 4, pp. 87-156.

QUINN, N. R. 1975. The placement problem as viewed from the physics of classical mechanics. In Proceedings of the 12th Design Automation Conference. pp. 173-178.

QUINN, N. R., AND BREUER, M. A. 1979. A force directed component placement procedure for printed circuit boards. IEEE Trans. Circuits Syst. CAS-26 (June), 377-388.

RANDELMAN, R. E., AND GREST, G. S. 1986. N-city traveling salesman problem: Optimization by
simulated annealing. J. Stat. Phys. 45, 885-890.

ROBSON, G. 1984. Automatic placement and routing of gate arrays. VLSI Design 5, 4, 35-43.

ROMEO, F., AND SANGIOVANNI-VINCENTELLI, A. 1985.Convergence and finite time behavior of simulated annealing. In Proceedings of the 24th Conference on Decision and Control. pp. 761-767.

ROMEO, F., SANGIOVANNI-VINCENTELLI,A., AND SECHEN, C. 1984. Research on simulated annealing at Berkeley. In Proceedings of the IEEE International Conference on Computer Design. pp. 652-657.

SAHNI, S., AND BHATT, A 1980. The complexity of design automation problems. In Proceedings of
the 17th Design Automation Conference. pp. 402-411.

SANGIOVANNI-VINCENTELLI,A. 1987. Automatic layout of integrated circuits. In Design Systems for VLSI Circuits, G. De Micheli, A. Sangiovanni-Vincentelli, and P. Antognetti, Eds. Kluwer Academic Publishers, Hingham, Mass., pp. 113-195.

SCHWEIKERT, D. G. 1976 A 2-dimensional placement algorithm for the layout of electrical circuits. In Proceedings of the Design Automation Conference. pp. 408-416.

SCHWEIKERT, D. G., AND KERNIGHAN, B. W. 1972. A proper model for the partitioning of electrical
circuits, In Proceedings of the 9th Design Automation Workshop. pp. 57-62.

SECHEN, C. 1986. The TimberWolf3.2 Standard Cell Placement and Global Routing Program. User's Guide for Version 3.2, Release 2.

SECHEN, C. 1988a. Chip-planning, placement, and global routing of macro/custom cell integrated circuits using simulated annealing. In Proceedings of the Design Automation Conference. pp. 73-80.

SECHEN, C. 1988b. VLSI Placement and Global Routing Using Simulated Annealing. Kluwer, B. V., Deventer, The Netherlands.

SECHEN, C. AND LEE, E .-W. 1987. An improved simulated annealing algorithm for row-based placement. In proceedings of the IEEE International Conference on Computer-Aided Design. pp. 478-481.

SECHEN, C., AND SANGIOVANNI-VINCENTELLI, A. 1986. TimberWolf3.2: A new standard cell placement and global routing package. In Proceedings of the 23rd Design Automation Conference. pp. 432-439.

SHA, L. AND BLANK, T. 1987. ATLAS: A technique for layout using analytic shapes. In Proceedings of the IEEE International Conference on Computer-Aided Design. pp. 84-87.

SHA, L. AND DUTTON, R. 1985. An analytical algorithm for placement of arbitrarily sized rectangular
blocks. In Proceedings of the 22nd Design Automation Conference. pp. 602-607.

SHAHOOKAR, K., AND MAZUMDER,P. 1990. A genetic approach to standard cell placement using meta-genetic parameter optimization. IEEE Trans. Comput.-Atded Design 9, 5 (May), 500-511.

SHIRAISHI, H. AND HIROSE, F. 1980 Efficient placement and routing techniques for masterslice LSI In Proceedings of the 17th Design Automation Conference. pp. 458-464.

SOUKUP, J. 1981, Circuit layout. Proc. IEEE 69, 10(Oct,), 1281-1304.

STEINBERG, L 1961 The backboard wiring problem: A placement algorithm. SIAM Rev. 3, 1 (Jan.), 37-50.

STEVENS, J. E. 1972. Fast heuristic techniques for placing and wiring printed circuit boards. Ph.D. dissertation, Comp. Sci. Dept., Univ. of Illinois.

SUARIS, P , AND KEDEM, G. 1987. Quadrisection: A new approach to standard cell layout In Proceedings of the IEEE International Conference on Computer-Aided Deszgn. pp. 474-477

SZU, H. 1986. Fast simulated annealing. In Proceedings of the AIT Conference. Neural Networks for Computmg, pp. 420-425.

TSAY, R., KUH, E. AND HSU, C. 1988. Module placement for large chips based on sparse linear
equations. Int. J Circuit Theory Appl. 16, 411-423.

UEDA, K., KASAI, R., AND SUDO, T. 1986 Layout strategy, standardization, and CAD tools. In Layout Design And Verificatton, T. Ohtsuki, Ed. Elsevier Science Pub. Co., New York, Chap. 1.

VECCHI, M. P., AND KIRKPATRICK, S. 1983. Global wiring by simulated annealing IEEE Trans, Comput.-Aided Design CAD-2, 215-222.

VLSI SYSTEMS DESIGN STAFF. 1987. Survey of automatic layout software. VLSI Syst. Design 8, 4, 78-89.

VLSI SYSTEMS DESIGN STAFF. 1988. Survey of automatic IC layout software. VLSI Syst. Design 9, 4, 40-49

WALSH, G. R. 1975. Methods of Optimization John Wiley and Sons, New York.

WHITE, S. R. 1984. Concepts of scale in simulated annealing In Proceedings of the IEEE International Conference on Computer Design. pp. 646-651

WIPFLER, G. J., WIESEL, M., AND MLYNSKI, D. A. 1982 A combined force and cut algorithm for hierarchical VLSI layout. In Proceedings of the 19th Design Automation Conference. pp. 671-677.

WONG, D F., LEONG, H. W , AND LIU, C. L 1986. Multiple PLA folding by the method of simulated annealing In proceedings of the Custom IC Conference. pp. 351-355.

WONG, D. F,, LEONG, H. W., AND LIU, C, L. 1988.Placement. In Simulated Annealing for VLSI Design, Kluwer B. V., Deventer, The Netherlands, Chap. 2.

CMOS VLSI Design: A Circuits and Systems Perspective, 4/E, Neil Weste, David Harris Publisher: Addison-Wesley Copyright: 2011

Logical Effort: Designing Fast CMOS Circuits (The Morgan Kaufmann Series in Computer Architecture and Design) Ivan Sutherland Robert F. Sproull David Harris

Digital Integrated Circuits (2nd Edition) Jan M. Rabaey, Anantha Chandrakasan , Borivoje Nikolic

A Tutorial on HSPICE, Owen Casha B. Eng. (Hons.) – 2005 [7] High Speed CMOS Circuit Design (c) 1997 David Harris Lecture 2: Static Design

UNIVERSITY OF CALIFORNIA AT BERKELEY College of Engineering Department of Electrical Engineering and Computer Sciences EE105 Lab Experiments HSPICE Tutorial

Notes on Library Characterization, Cell Characterization and Abstract Genenration , Yu Long

Standard Cell Characterization, Richard Sohnius, Computer Architecture Group Prof. Dr. U. Brüning University of Mannheim 16.12.2003 [12]Introduction to HSpice, Dr.-Ing. Frank Sill Department of Electrical Engineering, Federal University of Minas Gerais, Av. Antônio Carlos 6627, CEP: 31270-010, Belo Horizonte (MG), Brazil

B. Halpin, C. R. Chen, and N. Sehgal. Timing driven placement using physical net constraints. In Proc. Design Automation Conf., pages 780–783, 2001.

K. Rajagopal, T. Shaked, Y. Parasuram, T. Cao, A. Chowdhary, and B. Halpin. Timing driven force directed placement with physical net constraints. In Proc. Int. Symp.on Physical Design, pages 60–66, 2003.

S. Hur, T. Cao, K. Rajagopal, Y. Parasuram, A. Chowdhary, V. Tiourin, and B. Halpin. Force directed mongrel with physical net constraints. In Proc. Design Automation Conf., pages 214–219, 2003.

S. Hur and J. Lillis. Mongrel: Hybrid techniques for standard cell placement. In Proceedings of the International Conference on Computer-Aided Design, pages 165–170. IEEE, 2000.

B. Halpin, C. Y. R. Chen, and N. Sehgal. Detailed placement with net length constraints. In Proc. The 3rd International Workshop System on Chip, page 22, 2003.

M. A. B. Jackson and E. S. Kuh. Performance-driven placement of cell based ic's. In Proc. Design Automation Conf., pages 370–375, 1989.

A. Srinivasan, K. Chaudhary, and E. S. Kuh. Ritual: A performance driven placement algorithm for small cell ics. In Proc. Int. Conf. on Computer Aided Design, pages 48–51, 1991.

W. Swartz and C. Sechen. Timing driven placement for large standard cell circuits. In Proc. Design Automation Conf., pages 211–215, 1995.

A. Chowdhary, K. Rajagopal, S. Venkatesan, T. Cao, V. Tiourin, Y. Parasuram, and B. Halpin. How accurately can we model timing in a placement engine. In Proc. Design Automation Conf., pages 801–806, 2005.

T. Luo, D. Newmark, and D. Z. Pan. A new LP based incremental timing driven placement for high performance designs. In Proc. Design Automation Conf., 2006.