UNIVERSITÀ DEGLI STUDI DI PISA

FACOLTÀ DI INGEGNERIA

Corso di laurea in

Ingegneria Elettronica

Tesi di laurea

SVILUPPO E SPERIMENTAZIONE DI UN ALGORITMO EFFICIENTE DI UNO SCHEMA DI PALLETTIZZAZIONE PER UN ROBOT ANTROPOMORFO UTILIZZATO PER LA COSTRUZIONE DI PANCALI MONOPRODOTTO

Il candidato: Boschi Massimiliano

I relatori : Prof. Antonio Bicchi
Prof. Aldo Balestrino
Ing. Denis Mattia De Micheli
Prof. Ottorino Bruno

Anno Accademico 2004-2005

Indice dei Capitoli

Indice delle Figure

Abstract[1]

The PLP (Pallet Loading Problem) is the problem of packing small identical items (i.e. boxes or products) into a large object (the pallet).

Similar applications interest many areas as warehousing and distribution. Recent rises in transportation and storage costs have highlighted the importance to optimize the space and the time in the packing process. This means: maximizing the number of products that can be fitted on the pallet, maximizing the load stability, and minimizing the time to complete the packing process. Unfortunately these three procedures are not independent each other and the best solution could change in dependence of the necessity of the user, and in dependence of the automation system employed in the process.

In this paper we try to consider the whole PLP problem, realized by a anthropomorphic robot, in order to give a solution that could satisfy the real necessity of the user.

According to the typology of Dyckhoff, this is a problem of type 2/B/O/C.

[1] Parole chiave : algorithm, problem, loading, pallet, PLP, efficient, automated, order, pick, stability

Introduzione

Identificazione del problema

Il problema in questione è il seguente: dato un pallet e un certo prodotto da imballare su di esso mediante un sistema pallettizzatore con robot antropomorfo, stabilire la configurazione più efficiente per l'imballaggio. Questo è un tipico problema di stoccaggio che molte aziende operanti nel settore si trovano a dover affrontare. Ovviamente l'idea è quella di minimizzare i costi di produzione; a tal fine si deve cercare di riempire il pallet con il maggior numero di prodotti, impiegando il minor tempo possibile. Tale problematica rientra nei più generali problemi di C&P (Cutting and Packing) nei quali occorre riempire un volume con un certo numero di oggetti. Il problema è stabilire la configurazione finale che tali oggetti avranno in questo volume e il modo in cui dovranno essere disposti in esso. Il collocamento dei prodotti viene effettuato da un sistema automatizzato che li preleva dal fine linea. A tal fine occorre stabilire sia la collocazione spaziale di ogni oggetto, sia la sequenza temporale di prese e rilasci che dovranno essere effettuate. E' evidente che un simile problema è di natura 3D o anche 4D se consideriamo la variabile tempo; inoltre volendo considerare il caso più generale, dovremmo tenere in considerazione la possibilità che gli oggetti siano di varia natura e che eventualmente possano andare ad essere stoccati in recipienti di tipo diverso. Un simile approccio così generico, produrrebbe un problema di tipo NP, la cui risoluzione dovrebbe tener conto di troppe variabili e risulterebbe insolubile anche con l'ausilio dei più moderni processori. Per semplificare l'indagine si devono stabilire dei vincoli che vanno valutati in base al caso particolare da analizzare. Partendo da questo approccio, *Harald Dyckhoff*, in un articolo del 1990, *A typology of cutting and packing problems* [1], ha cercato di riclassificare in modo più dettagliato il generico C&P problem. Nel citato articolo è stata elaborata una terminologia specifica che è ormai considerata standard e viene comunemente utilizzata in letteratura per identificare un particolare problema di C&P. La visione di Dyckhoff parte dall'idea che il volume

da riempire possa essere considerato come un oggetto contenitore (large object) che va appunto riempito con oggetti più piccoli (small items).

In questo modo il problema può essere classificato in base a:

1. Spazio dimensionale del problema che dipenderà dalle forme geometriche del contenitore e degli oggetti. In questo modo lo spazio dimensionale potrà essere di tipo

   1D

   2D

   3D

   …

   ND (più in generale)

2. Tipo di assegnamenti tra contenitore e oggetti

   B ('Beladeproblem' termine tedesco che indica come una selezione di oggetti in questione debba essere combinata per il riempimento dei contenitori)

   V ('Verladeproblem' termine tedesco che indica come tutti gli oggetti in questione vadano assegnati a determinati contenitori)

3. Assortimento del contenitore: potrà essere unico, o potranno essercene diversi e di diverso tipo

   O (One object)

   I (Identical figure)

   D (Different figures)

4. Assortimento degli oggetti: molti, pochi, di natura uguale o diversa

   F (Few items of different figures)

M (Many items of many different figures)

R (many items of Relatively few different (non-congruent) figures)

C (Congruent figures)

Secondo questa terminologia, il nostro problema in questione appartiene alla categoria 2/B/O/C.

Infatti, occupandoci di imballaggi monoprodotto cioè con tutti gli oggetti uguali, cambierà soltanto la loro orientazione: siamo nel caso di oggetti congruenti (Congruent figures). Dal momento che tutti i pallet verranno assemblati nello stesso modo, sarà sufficiente analizzare il riempimento di un singolo pallet (One large object). A priori non viene stabilito il numero di oggetti da fissare sul pallet, ma dobbiamo scegliere la configurazione che inserisca il maggior numero di oggetti nell' imballaggio, per questo il problema può essere considerato un problema di tipo Beladeproblem. Inoltre, dato che gli oggetti sono tutti uguali, alla fine il pancale completo sarà realizzato sovrapponendo più strati di oggetti. In questo modo il tutto si riduce a determinare la composizione dei vari strati. Essendo i prodotti di nostro interesse approssimabili a forma di parallelepipedo, il singolo strato potrà essere analizzato considerando come riempire la superficie del pallet con le superfici di base dei prodotti: detto in altri termini, la soluzione può essere ricondotta alla determinazione di come riempire un grande rettangolo (superficie di base del Pallet) con rettangoli più piccoli (superficie d'appoggio degli oggetti) che è ovviamente un problema di tipo 2D.

Un simile problema, con i vincoli suddetti, viene comunemente definito PLP: Pallet Loading Problem.

Un problema di tipo NP

Una volta stabilito il tipo di robot da utilizzare, possiamo pensare di suddividere il processo di pallettizzazione in tre fasi principali:

1. riempimento del volume sul Pallet
2. realizzazione di una pallettizzazione con una buona stabilità
3. miglior utilizzo possibile della linea di trasmissione

Il riempimento del volume sul pallet mira a massimizzare la quantità di prodotti che è possibile imballare. Realizzare un algoritmo che determini la configurazione degli strati contenenti il numero massimo di pezzi pallettizzabili è quindi di primaria importanza.

Tuttavia la determinazione di uno strato che contenga il numero massimo di pezzi, non risolve il problema. Gli strati devono appoggiare l'uno sull'altro e dunque occorre fare in modo che la pila che va a formarsi risulti stabile. Oltretutto si deve tenere in considerazione che il pallet, una volta costruito, dovrà essere spostato quindi la pila di strati sarà soggetta anche ad eventuali forze dinamiche.

Infine, una volta stabilite le possibili configurazioni stabili, sarà necessario valutare qual è quella che il nostro sistema automatizzato è in grado di realizzare più velocemente, ossia quella in cui il robot adibito alla pallettizzazione riesca a sfruttare al meglio la linea di trasmissione. Ciò dipenderà dal sistema automatizzato scelto.

Realizzare una pallettizzazione ottima secondo questi tre criteri ci permette di ridurre costi e tempi di produzione.

Nonostante i vincoli introdotti e le conseguenti semplificazioni che si ottengono rispetto ad un più generico C&P problem, il PLP rimane di tipo NP-hard, ossia non conosciamo alcun algoritmo che possa risolvere in modo efficiente qualsiasi istanza. Come vedremo già il riempimento del volume sul pallet è un NP-hard e nel nostro caso la complessità aumenta ulteriormente in quanto le soluzioni di questo sottoproblema andranno ulteriormente combinate insieme per poterne valutare

gli effetti della stabilità e dell'utilizzo della linea. Un simile problema ammette un numero esponenziale ma finito di soluzioni, quindi in linea di principio, potremmo pensare di enumerare tutte le soluzioni, valutarne il costo e scegliere la migliore. Tale approccio si rivela però impraticabile, in quanto troppo oneroso dal punto di vista computazionale. La strada da seguire dovrà essere quella di limitare le soluzioni da analizzare intuendo quali siano quelle più adatte al nostro scopo. Andranno stabilite delle opportune regole di taglio, che il sistema automatizzato dovrà applicare autonomamente o con l'aiuto dell'utente, per arrivare ad una soluzione quanto meno accettabile.

Un prodotto versatile

Data la complessità del problema in esame, è quasi impossibile riuscire a prevedere tutte le casistiche che l'algoritmo elaborato si troverà a dover affrontare. I parametri che descrivono il problema possono essere variabili e per di più soggetti alle particolari esigenze pratiche. Di conseguenza, in molti casi potrebbe essere gradita un'interazione diretta con l'utente esterno. In generale il programma dovrà essere in grado di fornire una soluzione partendo solo dall'inserimento dei dati. In certe situazioni però, si potrebbe pensare di semplificare il lavoro del software andando a favorire dei tagli gestiti direttamente dall'esterno. In questo modo si eviterebbe ad esempio di valutare numerose soluzioni che un utente esperto potrebbe intuire di voler escludere a priori. Nell'ottica di realizzare un prodotto commerciale di questo genere, l'interazione con l'utente andrà sempre opportunamente valutata.

Implementazione

In simili applicazioni, la programmazione orientata ad oggetti mostra tutta la sua versatilità ed efficienza: per tali motivi il Visual C++ 6.0 è stato scelto come linguaggio di programmazione.

Il programma che è stato così realizzato, si chiama PALLETTIZZA.

Capitolo 1 : Scelta del sistema automatizzato da adottare

1.1. Pallettizzatore con robot antropomorfo

I sistemi pallettizzatori attualmente disponibili offrono innumerevoli vantaggi rispetto ad una pallettizazzione realizzata in maniera manuale, basti pensare alle problematiche che si avrebbero relativamente a carichi pesanti, spazi di lavoro ampi, spreco di manodopera e conseguentemente costi elevati. Tali impianti automatizzati si distinguono in due macro tipologie: pallettizzatori cartesiani o a portale e pallettizzatori con robot antropomorfo. I primi, vedi Figura 1.1, sono principalmente utilizzati per la pallettizzazione dove non sia richiesta particolare flessibilità; i secondi invece, vedi Figura 1.2, vengono utilizzati in applicazioni estremamente sensibili, quando le esigenze di pallettizzazione mutano velocemente al variare delle esigenze del cliente, quando servono sistemi veloci, affidabili e per cicli produttivi intensi. I robot antropomorfi garantiscono prestazioni nel tempo, velocità, ripetibilità, affidabilità, su turni di 24 ore, difficilmente raggiungibili dai sistemi cartesiani. Grazie all'elevata flessibilità di un robot antropomorfo, il sistema è in grado di pallettizzare prodotti singoli, multipli e carichi misti con schemi di impilamento molto complessi e su più pallet.

Figura 1.1: pallettizzatore cartesiano o a portale

Figura 1.2: pallettizzatore con robot antropomorfo

Gli impianti integrati con robot antropomorfi rappresentano dunque la spinta per una nuova evoluzione industriale. Grazie al loro superiore campo di lavoro ed al tipico cinematismo articolato consentono la realizzazione di impianti innovativi e flessibili, con un alto numero di operazioni. Si osservi ad esempio una tipica isola di lavoro rappresentata in Figura 1.3. La struttura modulare aperta risulta flessibile e adattabile alle varie lavorazioni: forte di un ampio volume di lavoro a forma sferica, il sistema può ottenere una estrema libertà di configurazione. Da questo punto di vista l'uso di un robot antropomorfo permette un miglior utilizzo dello spazio di lavoro rispetto ai concorrenti cartesiani. L'elevata capacità negli spazi ridotti è un altro punto di forza. Con una adeguata programmazione possono essere gestiti automaticamente non solo il cambio di pallet, fogli di interfalda, ma anche tutte le varianti di prodotto. Inoltre si possono gestire carichi misti, pallet multipli o parziali e governare stazioni di pallettizzazione alimentate da più sistemi di trasporto. Possiamo pertanto affermare che, sebbene nel caso di semplici operazioni i robot cartesiani siano ancora competitivi rispetto ad un robot antropomorfo, in situazioni complesse l'utilizzo di un'isola

robotizzata integrata con stazioni multiple di lavorazione risulta sicuramente la soluzione più affidabile e conveniente.

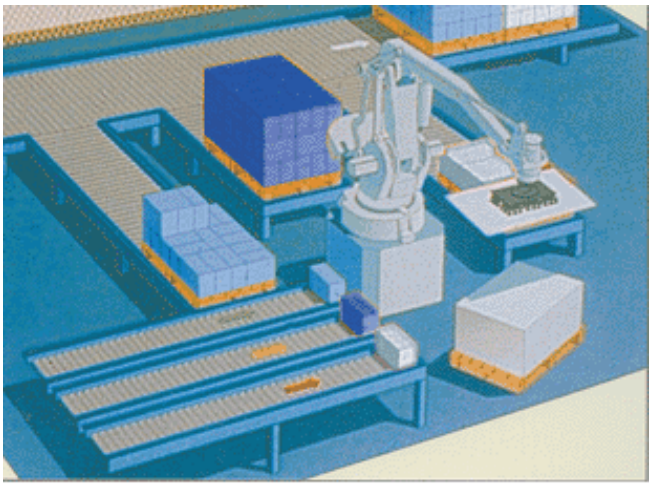


Figura 1.3: isola di lavoro per un pallettizzatore con robot antropomorfo

Volendo realizzare un prodotto versatile, la scelta dovrà necessariamente cadere sull'utilizzo di un pallettizzatore dotato di robot antropomorfo.

1.2. Vasta gamma di sistemi di presa

Grazie ai diversi tipi di end-effector a disposizione, il sistema di pallettizzazione può adattarsi alle più svariate tipologie di prodotti di peso e misura variabile, sia rigidi che fardellati. A seconda del prodotto in questione, dovranno essere utilizzate prese ad aspirazione, con pinze meccaniche o anche magnetiche.

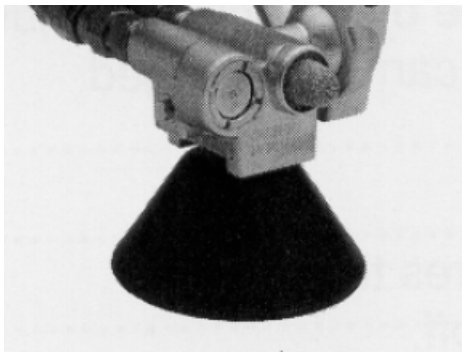


Figura 1.4: end effector a ventosa

La Figura 1.4 mostra un tipico end-effector a ventosa particolarmente efficace per prodotti rigidi e non particolarmente pesanti. Per fardelli sempre non troppo pesanti, come ad esempio pacchi di pasta per l'industria alimentare, può essere utilizzato un organo terminale ad aspirazione. Per oggetti di dimensioni e pesi maggiori, solitamente vengono impiegate delle pinze tipo quelle illustrate in Figura 1.5. Il loro funzionamento pneumatico a pressione controllata, consente di prelevare delicatamente i colli garantendo l'integrità del prodotto.

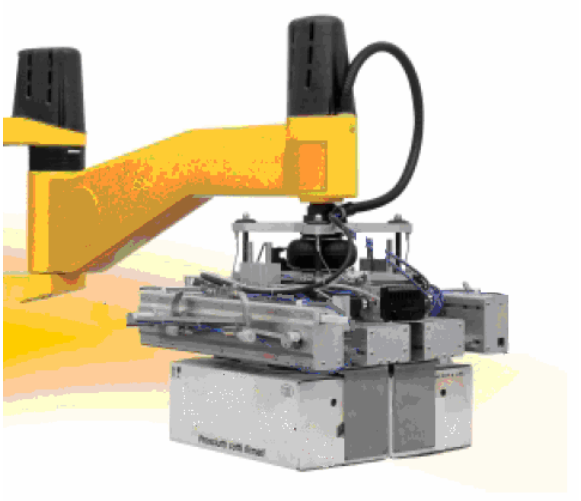


Figura 1.5: pinze attuate

Per i fardelli invece sono utilizzate delle pinze a forma più avvolgente come in Figura 1.6. In questo modo si riesce a sostenere il sacco anche sulla parte inferiore. Data la loro particolare forma, sono definite pinze di prelievo a forca.

Figura 1.6: pinze di prelievo a forca

Il pallettizzatore deve essere in grado di gestire sia i prodotti, sia l'interfalda, sia il pancale stesso. La strategia più utilizzata è quella di montare sul suo end-effector tutti gli organi necessari. Così sulla parte inferiore solitamente sono collocate le pinze per la manipolazione dei prodotti; più esterne sono inserite delle pinze a forca per agganciare il pancale; infine possono anche essere presenti degli attuatori ad aspirazione per la manipolazione delle interfalde, vedi Figura 1.7. Quando agisce un organo, gli altri vengono opportunamente retratti in modo da non interferire.

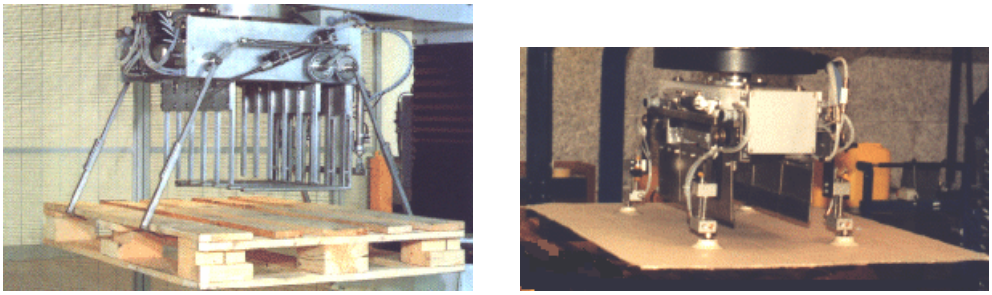


Figura 1.7: end-effector in grado di manipolare più oggetti diversi tra loro

1.3. Linea di trasmissione

Gli end-effectors più moderni possono gestire prese multiple, ossia sono in grado di prelevare e rilasciare più prodotti con un unico movimento. Ciò è possibile grazie all'attuazione differenziata dei vari organi di presa componenti la pinza. Affinché il robot possa afferrare più prodotti insieme, è però indispensabile che tali prodotti siano stati opportunamente accumulati e preparati al fine linea. La Figura 1.8 mostra due esemplificazioni delle più comuni linee di trasporto a rulli o a nastro.

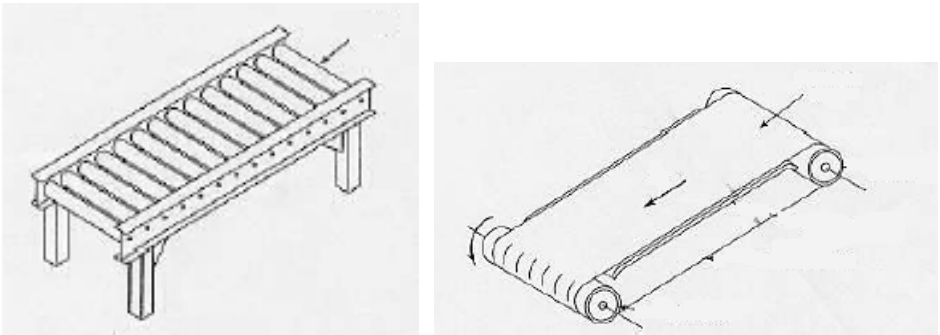


Figura 1.8: convogliatori a rulli e a nastro

I trasportatori a rulli sono particolarmente indicati quando si devono realizzare degli accumuli di prodotti. Molte aziende produttrici infatti si sono specializzate in sistemi a rulli frizionati che, in presenza di accumuli, si fermano riducendo la spinta fra un prodotto e l'altro, evitando così danneggiamenti o schiacciamenti fra colli. Il trasportatore accumula i prodotti sempre nelle stesso verso, tuttavia è possibile ottenere l'accumulo di prodotti ruotati mediante l'utilizzo di opportuni sistemi di rotazione. Un esempio è fornito dai trasferitori ortogonali che sono utilizzati quando si deve deviare ortogonalmente il prodotto.

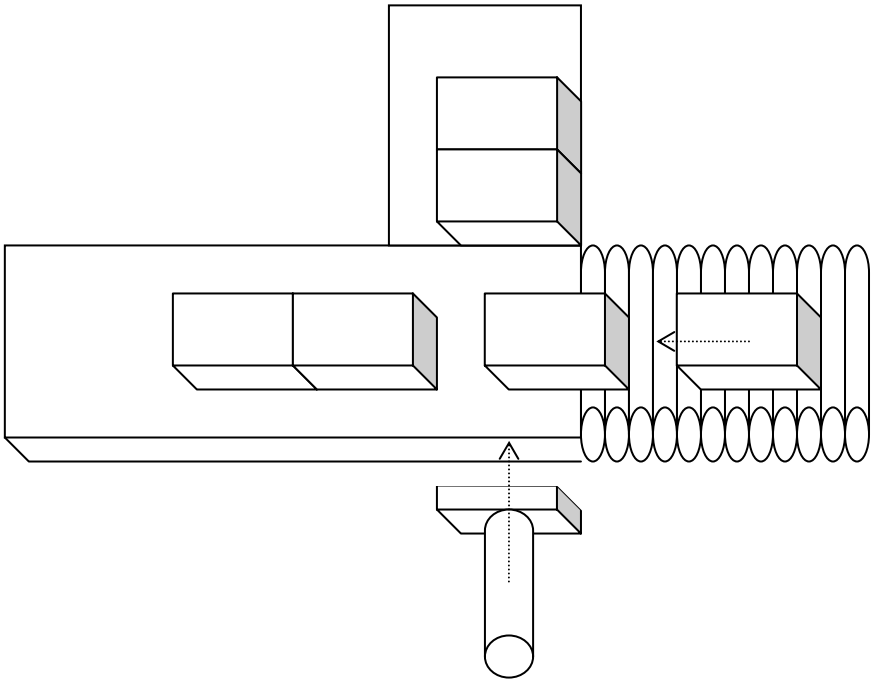


Figura 1.9: schema di un trasferitore ortogonale

Un esempio banale di trasferitore ortogonale è fornito in Figura 1.9: i prodotti vengono accumulati in un verso dal nastro trasportatore, ma possono essere accumulati in maniera ruotata semplicemente spingendoli lateralmente. Trasferitori più complessi sono costituiti da dei blocchi montati lungo la linea di trasporto come quello illustrato in

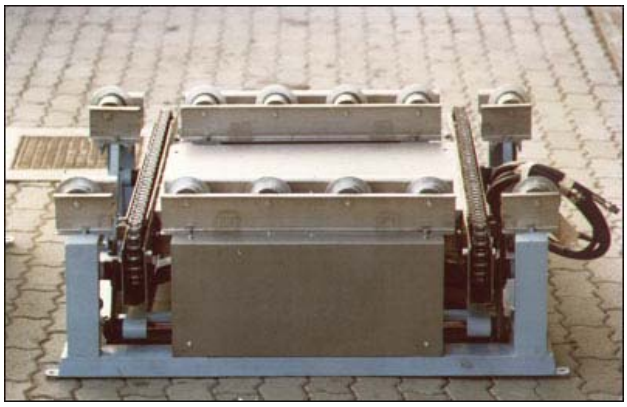


Figura 1.10: trasferitore ortogonale

non appena il collo passa sopra di esso viene sollevato sopra il piano e staccato dal trasportatore sul quale transitava, quindi si comanda la rotazione mediante un gruppo di catene e si riabbassa quando si è raggiunta l'orientazione desiderata.

1.4. Conclusioni

In definitiva, mediante l'utilizzo di un robot antropomorfo è possibile riuscire a gestire la quasi totalità dei prodotti presenti sul mercato. Per ragioni pratiche ci limiteremo a considerare i soli quei prodotti riconducibili in prima approssimazione ad una forma a parallelepipedo, siano essi prettamente rigidi o fardellati.

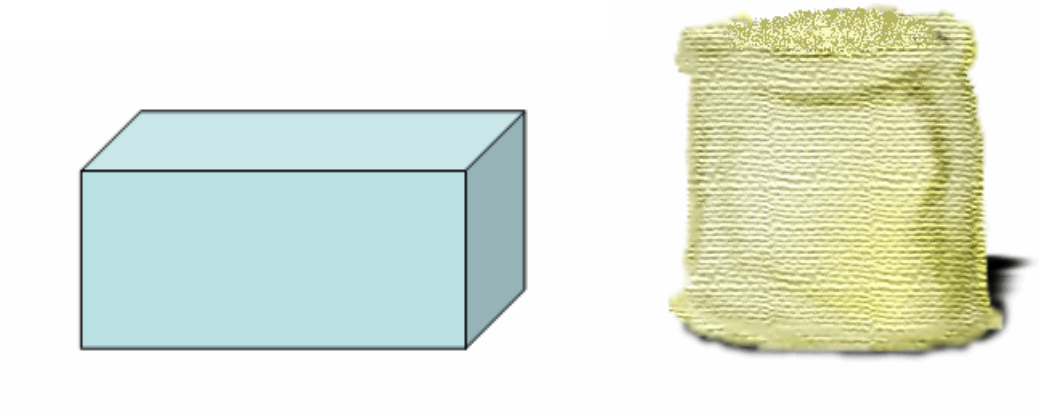


Figura 1.11: prodotti in esame

A questo punto, il problema è quello di elaborare un algoritmo che ottimizzi il modo di distribuire tali prodotti sul pallet.

Capitolo 2 : Massimizzazione del numero di prodotti pallettizzabili

2.1. Pallet formato dalla sovrapposizione di più strati

Figura 2.1: un pallet completo

La massimizzazione del numero di pezzi all'interno di un pallet completo è quasi sempre considerato un punto di primaria importanza. Riuscire ad imballare più prodotti possibili su un unico pallet, permette di realizzare un guadagno di spazio facilitando i trasporti con conseguente diminuzione dei costi di produzione. Consideriamo il pallet completo come la sovrapposizione di più strati di prodotti: massimizzando il numero di prodotti all'interno dei singoli strati avremo massimizzato anche il numero complessivo di prodotti all'interno dell'intero imballaggio. Vedremo poi dalle successive analisi, che nel nostro caso il problema sarà quello di determinare due soli strati tali che, combinati insieme, risultino la coppia più efficiente per la pallettizzazione dal punto di vista dell'occupazione del volume, della stabilità e dell'utilizzo della linea di trasmissione. Quindi il

nostro sottoproblema di partenza sarà quello di determinare gli eventuali layouts degli strati che inseriscono il maggior numero di prodotti al loro interno. Come già detto, dato che i prodotti di nostro interesse possono essere approssimati con forma a parallelepipedo, la valutazione del singolo strato potrà essere analizzata considerando come riempire la superficie rettangolare di base del Pallet con le superfici d'appoggio dei prodotti, anch'esse rettangolari. In questo modo ci riconduciamo ad un problema di tipo 2D. Detto in altri termini si ha il problema geometrico di ricoprire la superficie di un grande rettangolo con rettangoli più piccoli. Nonostante le semplificazioni così ottenute, questo sottoproblema, anche se non è stato rigorosamente dimostrato, viene comunque considerato di tipo NP-hard perché le combinazioni che possono scaturire sono praticamente incalcolabili già per casi di medie dimensioni. Le sue soluzioni sono state ampiamente studiate in letteratura fin dalla fine degli anni '70 e nel corso del tempo sono stati prodotti vari algoritmi che cercano di risolverlo, sia in modo euristico che in modo esatto.

2.2. Piazzamento ortogonale dei pezzi

Per ragioni pratiche tutti gli algoritmi che tratteremo prendono sempre in considerazione soltanto configurazioni con i pezzi piazzati ortogonalmente, ovvero con i lati dei prodotti paralleli ai lati del pallet. Questo ovviamente semplifica l'algoritmo da utilizzare che altrimenti diverrebbe ingestibile a causa dell'esplosione di tutte le possibilità che andrebbero valutate. Teoricamente ci sono anche dei casi in cui un layout con piazzamento non ortogonale, potrebbe effettivamente riuscire a ridurre l'area inutilizzata sul pallet, ottenendo un miglioramento sul rapporto volume occupato su numero di pezzi inseriti, rispetto a qualunque altro layout ortogonale. Tuttavia simili situazioni sono delle eccezioni che interessano per lo più particolari casi degeneri, come i prodotti a base quadrata [30] [31] [32], ma non coinvolgono quasi mai la realtà pratica dei più comuni sistemi di pallettizzazione. Innanzi tutto il sistema automatizzato dovrebbe essere in grado di posizionare i pezzi con qualunque rotazione, ma questo non dovrebbe essere difficile per i più moderni sistemi di automazione. Il problema maggiore è che progettare un algoritmo in grado di risolvere anche i layouts non

ortogonali porterebbe decisamente ad un aumento dei tempi computazionali, e i risultati forniti, con molta probabilità, finirebbero per risultare significativi solo in casi troppo sporadici per poter compensare il tempo effettivamente perso in tutte le altre casistiche. Inoltre si deve valutare il fatto che il processo di pallettizzazione dovrà tenere in considerazione non soltanto la massimizzazione del numero di prodotti da impacchettare, ma anche la stabilità complessiva del sistema e il tempo necessario per realizzare il pallet completo. Nell'ottica della stabilità i layout non ortogonali potrebbero porre qualche problema agli angoli del pallet dove quasi sempre è consigliabile che vi sia un riempimento. Per quanto riguarda il problema dei tempi di composizione del pallet completo, esso è inevitabilmente collegato al sistema di prese-rilasci per la realizzazione dello strato: se i pezzi risultano disposti in modo sparpagliato, alcuni con una orientazione, altri con un'altra, sarà difficile trovare delle adiacenze che permettano di piazzare più pezzi con un'unica presa-rilascio, e ciò andrà inevitabilmente a influire sui tempi di completamento del pallet. Di conseguenza imbarcarsi in un'analisi analitica dei layouts non ortogonali, salvo in casi eccezionali, per il momento, sembra improduttivo. Eventuali aggiustamenti non ortogonali potrebbero essere effettuati da un operatore esperto che interagisca direttamente con il programma di pallettizzazione.

2.3. Algoritmi per la ricostruzione degli strati: stato dell'arte

Il problema di trovare le configurazioni con il numero massimo di pezzi inseribili in uno strato è stato ampiamente affrontato in letteratura. Solitamente viene identificato con una quaterna di numeri: [Y_pal , X_pal , Y_box , X_box] dove i primi due numeri esprimono le dimensioni del pallet mentre i restanti due le dimensioni del prodotto, quasi sempre espresse in mm. La soluzione dovrà fornire la lista dei pezzi indicando per ognuno di essi la sua collocazione spaziale sullo strato. Riferendoci a pallettizzazioni soltanto ortogonali, i pezzi potranno essere disposti esclusivamente con due orientazioni, orizzontale (H) o verticale (V), che differiscono l'una dall'altra per una rotazione di 90° del pezzo stesso. In questo modo la loro collocazione può essere espressa fornendo la posizione del baricentro o di uno dei quattro vertici del rettangolo di base rispetto al sistema di

riferimento, e la sua orientazione. Il problema così impostato è chiaramente di natura matematico-geometrica, tuttavia questo approccio non deve far perdere di vista le ulteriori esigenze di tipo pratico che si avranno per l'implementazione effettiva dell'intero processo di pallettizzazione. Il riempimento dello strato infatti non è l'unico problema: occorre valutare anche l'efficienza delle prese e la stabilità complessiva del sistema. In quest'ottica più strati diversi che riescano ad inserire lo stesso numero dei pezzi sul pallet, non potranno essere tutti considerati dello stesso valore e andranno rivalutati in base agli altri fattori di merito. Molti degli algoritmi che sono stati prodotti in letteratura mirano soltanto a risolvere il problema matematico-geometrico e non tengono in considerazione gli altri aspetti del processo di pallettizzazione. Gli algoritmi possono essere di due tipi: euristici o esatti.

2.3.1. Algoritmi Euristici

Gli algoritmi Euristici si basano su regole di natura pratico-intuitiva: non vengono analizzate tutte le possibili soluzioni bensì solo quelle con determinate caratteristiche. Andando ad analizzare i layout degli strati che vengono utilizzati per la pallettizzazione, si osserva che in molti casi è possibile ricondurli a degli schemi affini: si può quindi pensare di realizzare un algoritmo che faccia un analisi del layout seguendo questi schemi. E' appunto l'esperienza che suggerisce quali siano le caratteristiche da ricercare e quelle da scartare a priori. Tutto ciò porta inevitabilmente ad una limitatezza nei modelli di layout che potranno essere determinati dall'algoritmo. Questa limitatezza dei modelli risulta essere sia il loro punto di forza, perché li rende estremamente veloci in quanto non vengono ricercate tutte le soluzioni possibili, ma anche il loro punto di debolezza, perché se la soluzione ottimale non possiede quelle determinate caratteristiche prefissate, non potrà essere trovata dall'algoritmo. In genere buoni algoritmi euristici dovrebbero essere in grado di trovare la soluzione ideale per l'utente almeno nel 90% dei casi, risultato che sarebbe più che accettabile dal punto di vista pratico. Dalla fine degli anni '70 ad oggi, sono stati elaborati vari tipi di algoritmi

euristici che col passare del tempo sono stati sempre più affinati e completati per riuscire a raggiungere risultati sempre più efficienti.

Algoritmi Guillotine :

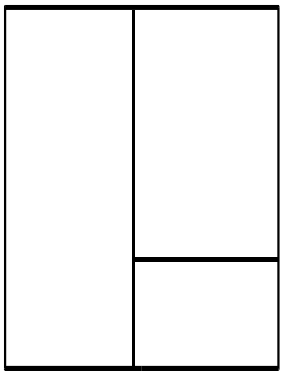


Figura 2.2: schema tipico di un layout realizzato con algoritmi guillotine

Sono i primi algoritmi di pallettizzazione ad essere stati prodotti [2]. L'idea di fondo è quella di dividere il layout del rettangolo di base con un taglio a ghigliottina (guillotine cut), ossia con un taglio dritto che va da un estremo all'altro del rettangolo in questione. In questo modo si ottengono altre due sezioni, sempre rettangolari, che potranno essere analizzate in maniera ricorsiva con lo stesso procedimento. La pratica ha dimostrato che simili algoritmi non sempre riescono a determinare le soluzioni con il numero massimo dei pezzi inseribile, quindi già per quello che riguarda la massimizzazione del numero dei prodotti all'interno del pallet completo, non risultano molto affidabili. La loro inadeguatezza è principalmente dovuta alla povertà geometrica del metodo di ricerca del layout: può succedere, e non raramente, che i layouts in cui sono inseribili il numero massimo dei pezzi non possano essere suddivisi intagli a ghigliottina. In letteratura è piuttosto facile trovare vari esempi che mostrano l'inefficacia di simili algoritmi [4] [5] [8]. Infine, come vedremo nell'analisi della stabilità, può essere conveniente evitare layout con simili tagli in quanto possono risultare dannosi per la stabilità finale del pallet: ciò è ovviamente impossibile utilizzando questo

algoritmo, dato che per come è impostato, potrà produrre soltanto layout con tagli a ghigliottina. In conclusione quindi, questi algoritmi appaiono ormai inefficaci sotto più punti di vista.

Algoritmi a 4 Blocchi:

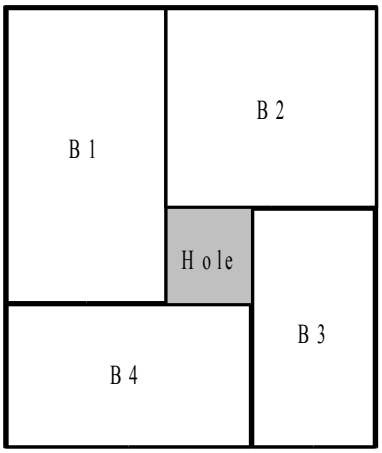


Figura 2.3: schema tipico di un layout realizzato con algoritmi a blocchi

Sono stati elaborati subito dopo gli algoritmi guillotine per cercare di superarne gli inconvenienti [4] [5]. L'idea base è piuttosto semplice: suddividere il layout in quattro sottoblocchi d'angolo. Tali sottoblocchi risultano formati da pezzi omogenei, ossia tutti con la stessa orientazione. Seguendo questa schematizzazione al centro del layout potrà formarsi un buco che se sufficientemente grande, dovrà essere riempito con ulteriori pezzi. L'algoritmo cerca la soluzione migliore al variare delle dimensioni dei blocchi d'angolo. Questo approccio risulta sufficientemente valido sia dal punto di vista dell'efficienza delle prese che dal punto di vista della stabilità. Per ciò che concerne le prese-rilasci infatti, la composizione a blocchi omogenei risulta particolarmente efficiente in quanto, avendo a disposizione un sistema automatizzato in grado di sistemare più pezzi contemporaneamente, risulta conveniente che i pezzi con la stessa orientazione siano adiacenti; per quanto riguarda la stabilità invece, una volta determinato un layout, potrebbe essere sufficiente alternarlo sui vari strati ruotato di 90° [6]. E' inutile dire che anche questi algoritmi molte volte mancano

il bersaglio di massimizzare il numero di pezzi inseribili nel layout, basta considerare gli esempi proposti in [9] [10]. Tuttavia il loro pregio principale è che, nella loro versione più semplice, possono essere facilmente implementati senza ricorsione e quindi risultano quasi istantanei.

Algoritmi a 5 Blocchi:

Sono la naturale evoluzione degli algoritmi a 4 Blocchi. L'idea di fondo è di valutare meglio il buco centrale che può formarsi considerandolo come un ulteriore blocco da analizzare [6]. Eventualmente si può anche riempire il buco centrale non come blocco omogeneo, ma riconsiderandolo come una sotto-istanza da ricomporre anch'essa a blocchi. Solitamente tale blocco centrale viene automaticamente adattato in dipendenza dei 4 blocchi d'angolo, e non viene considerato quindi come variabile. Tuttavia si potrebbe anche pensare di valutarlo come blocco autonomo in grado di espandersi e adattando conseguentemente i blocchi d'angolo: in questo modo si riuscirebbe ad aggiungere un altro grado di libertà all'algoritmo, riuscendo così ad ottenere ulteriori configurazioni diverse. Tutto questo comunque sempre a scapito della velocità d'implementazione.

Algoritmi di tipo G4:

Sono un'ulteriore evoluzione delle strutture a blocchi: ogni sottoblocco viene a sua volta considerato come una struttura a blocchi a se stante [9] [10]. La procedura implementata è ovviamente ricorsiva. Conseguentemente al crescere della complessità del problema si verifica un notevole peggioramento delle prestazioni in termini di velocità. Tuttavia dal punto di vista dei modelli ottenibili, questi algoritmi risultano molto versatili ed estremamente efficienti per problemi di piccole e medie dimensioni. Al crescere delle dimensioni del problema invece non è conveniente applicare direttamente l'algoritmo, ma si può cercare di abbassare la complessità computazionale suddividendo prima la superficie utile del pallet nei vari blocchi e applicando poi l'algoritmo ad ognuno di essi, come illustrato in [11]. In questo modo i G4 riescono a soddisfare anche problemi in cui si devono

piazzare più di un migliaio di pezzi continuando ad operare in tempi ragionevoli. Se si vuole individuare un limite nei layouts prodotti da questi algoritmi, lo si può trovare nel notevole intreccio di prodotti che si ha nelle soluzioni proposte. Come vedremo, nella maggior parte dei casi, questa caratteristica può costituire un vantaggio dal punto della stabilità finale, ma anche uno svantaggio per quanto riguarda l'efficienza delle prese-rilasci.

2.3.2. Algoritmi Esatti

Gli algoritmi esatti dovrebbero fornire tutti i possibili layouts diversi tra loro che comunque inseriscono lo stesso numero massimo di pezzi. Kathryn A.Downsland [12] ha cercato di risolvere questo problema associando al layout dello strato, un grafo che ha come vertici l'insieme di tutte le possibili posizioni dei pezzi dello strato: tali vertici risultano adiacenti se le superfici occupate dai rispettivi pezzi si sovrappongono. L'idea è quella di trovare un sottoinsieme dei vertici del grafo associato, tali che non ve ne siano di adiacenti. Teoricamente quindi, avvalendosi delle tecniche di studio dei grafi, è possibile determinare la soluzione. A tal fine potrebbe essere sufficiente consultare un qualunque testo sull'argomento: K.A.Dowsland cita Wilson R.J., *Introduction to graph theory* [13]. Il problema di questo approccio è che un PLP di medie dimensioni, finirebbe per vedersi associato un grafo con quasi un milione di vertici, la cui analisi risulterebbe troppo lunga anche per i più moderni processori. L'autrice ha quindi cercato di implementare questo metodo [14] applicando delle riduzioni che si basano sull'utilizzo di istanze equivalenti e operando con opportune regole di taglio che evitano di valutare tutti quei grafi i cui pezzi sono chiaramente sovrapposti. A tal fine ad esempio, può essere considerata la situazione lungo una linea verticale o orizzontale del pallet: se lungo tale linea, la somma delle lunghezze dei pezzi presenti su di essa, supera la dimensione del pallet, è ovvio che ve ne sono di sovrapposti. Un altro problema che si pone con un simile approccio, è relativo al numero di soluzioni che si possono ottenere. Consideriamo ad esempio la situazione illustrata in Figura 2.4:

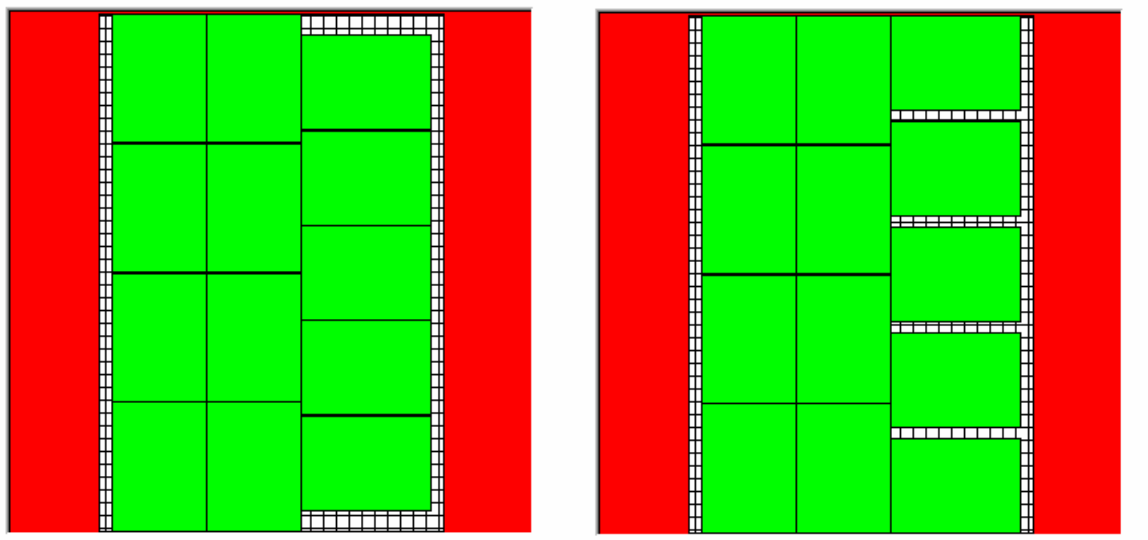


Figura 2.4: due soluzioni 'simili' per la stessa istanza [1200, 800, 300, 220]

L'istanza presa in considerazione è [1200, 800, 300, 220]: Le figure mostrano due possibili soluzioni che inseriscano entrambe il numero massimo di prodotti per strato. È evidente che il layout ricostruito è sempre lo stesso a meno di traslazioni dei pezzi incolonnati a destra. Ciò è dovuto al fatto che quando la superficie del piano del pallet non potrà sempre essere esattamente ricoperta con la superficie dei prodotti, si potrebbero generare dei buchi tali da permettere eventuali traslazioni di alcuni pezzi. Se non discretizziamo le posizioni possibili, al limite ci saranno infinite traslazioni con altrettanti layouts possibili. Ovviamente ciò non ha molto valore pratico in quanto il nostro sistema automatizzato piazzerà i prodotti con una certa precisione, quindi sarà sempre possibile discretizzare lo spazio a disposizione in base alle caratteristiche del sistema. Tuttavia anche in questo caso la situazione è praticamente ingestibile. Nell'esempio citato si verrebbero a creare tutte le possibili traslazioni di ogni singolo pezzo, combinate con tutte le possibili traslazioni degli altri pezzi. E' un vero e proprio calcolo combinatorio. Volendo esprimere le coordinate dei pezzi in mm, dato che sul pallet rimane un buco di 220mmx100mm, dovremmo considerare 100 possibili traslazioni per ogni singolo pezzo, e considerando che possono traslare contemporaneamente fino a cinque pezzi, ci si rende subito conto che non è possibile gestire tutte le soluzioni. Per evitare simili complicazioni, nella realizzazione di molti algoritmi esatti viene calcolato l'upper bound, ossia il numero massimo di pezzi teoricamente inseribili nel layout, per

bloccare l'algoritmo non appena viene trovata una configurazione con queste caratteristiche. Questo è quello che fa anche K.A.Dowsland nella sua implementazione pratica. Tuttavia questo approccio risolve esclusivamente il problema dell'occupazione del volume tralasciando tutte le altre questioni inerenti alla stabilità e alle prese-rilasci. In questo modo, alla fine si determinerebbe un solo layout, senza considerare che potrebbero essercene altri che inseriscono lo stesso numero di pezzi, ma con migliori caratteristiche sotto altri punti di vista. Di conseguenza per il nostro scopo, un buon algoritmo, più che trovare tutti i ridultati possibili, dovrebbe riuscire a valutare tutte le soluzioni base da cui possano poi scaturire le altre per eventuali traslazioni. Ovviamente una volta dato il layout base, si dovrà in qualche modo tenere conto di queste possibili traslazioni per poterne valutare gli effetti su prese e stabilità, in modo da poter ottimizzare la soluzione finale anche in relazione a questi parametri. In tale ottica, risulta estremamente interessante l'algoritmo proposto da Subir Bhatcharya, Rahul Roy e Sumita Bhatcharya [15]. Esso è basato su una ricerca ad albero di tipo MBFS (Maximal Breadth Filling Sequence), ossia si cerca sempre di massimizzare il numero di pezzi inseribili lungo una riga. Se consideriamo il seguente sistema di riferimento con origine su un angolo del pallet, illustrato in Figura 2.5

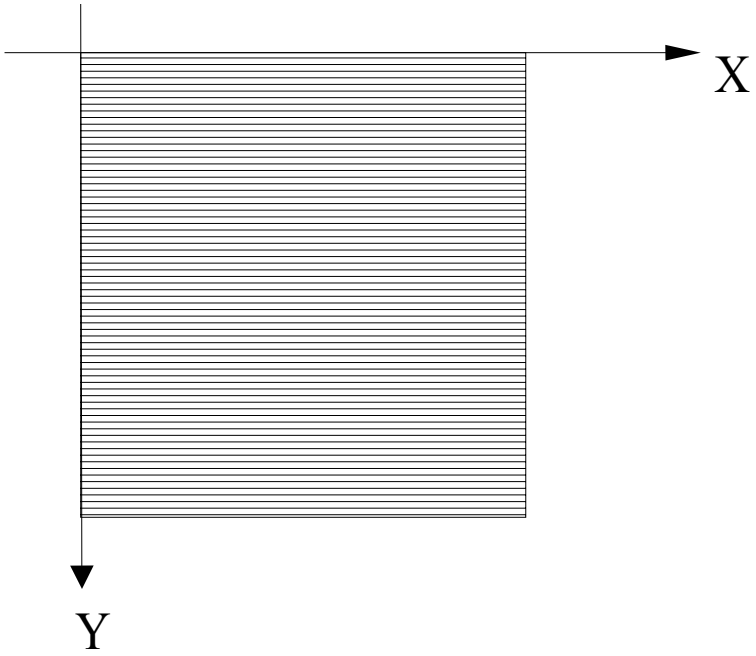


Figura 2.5: sistema di riferimento sul piano per l'identificazione della collocazione dei prodotti

L’algoritmo così come è impostato nell’articolo, per ogni ordinata riesce a fornire il numero di pezzi presenti lungo la linea orizzontale precisando quanti siano orizzontali e quanti verticali. Da questo punto di vista l’algoritmo non risolve ancora il nostro problema perché non fornisce la collocazione esatta dei pezzi, al più si può velocemente stimare la loro ordinata, ma non l’ascissa. Anzi potrebbero esserci più modi di disporre i pezzi lungo la linea: un semplice esempio è il solito caso in cui possano verificarsi delle possibili traslazioni orizzontali. Tuttavia a partire dalle informazioni generate, potremmo pensare di realizzare un altro algoritmo, da affiancare a questo, per una ricostruzione ad hoc dei layouts di nostro interesse.

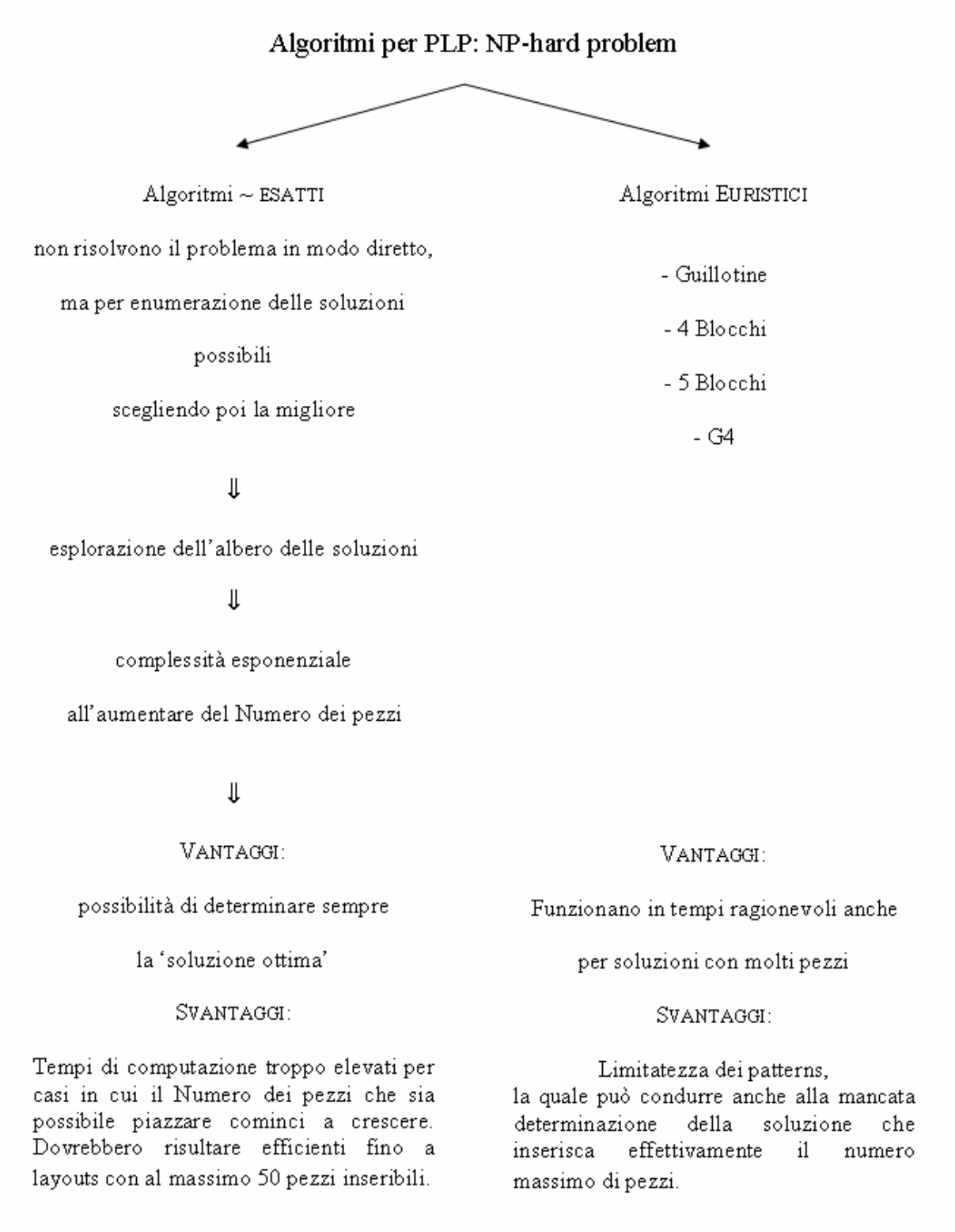


Figura 2.6: schema riassuntivo degli algoritmi analizzati per risolvere il PLP

2.3.3. Esatto o Euristico?

Fatta questa breve rassegna sui metodi utilizzati per cercare di ricostruire i layouts che inseriscano il numero massimo di pezzi, viene spontaneo chiedersi quale sia l'algoritmo da utilizzare. E' evidentemente che da una parte l'algoritmo esatto rischia di produrre troppe soluzioni per poter essere gestite, mentre dall'altra parte, l'utilizzo di un algoritmo euristico potrebbe condurre al problema opposto, ossia alla mancanza di possibili soluzioni che risultino utili al nostro scopo. Il problema degli euristici non è tanto nel numero ridotto delle soluzioni, in quanto come abbiamo detto sarebbe impossibile gestire tutte le soluzioni derivanti dal PLP, ma proprio nella loro qualità. Se riuscissimo a trovare un algoritmo euristico che risolvesse la maggioranza delle nostre applicazioni pratiche sarebbe l'ideale, in quanto guadagneremmo tempo evitando di dover analizzare situazioni tendenzialmente inutili. Nel nostro caso però, rivolgendoci ad una casistica piuttosto generale, sarà difficile che un euristico possa riuscire a soddisfare almeno il 90% dei casi di interesse. Il problema principale degli algoritmi euristici consiste nel fatto che trovano soltanto un certo numero di soluzioni, ma non è detto che siano tutte e solo quelle possibili, e cosa ancora più grave, non è neppure certo che siano soluzioni che inseriscono il numero massimo di pezzi. In effetti il problema di massimizzare il numero dei prodotti all'interno del layout è solitamente considerato di primaria importanza [20] e ha quasi sempre la precedenza sull'analisi delle altre due problematiche. E' infatti quasi scontato che la realizzazione di un pallet con più pezzi, solitamente conduca anche ad un vantaggio non solo di ingombro ma anche di tempo, in quanto l'eventuale tempo guadagnato dal mancato inserimento di alcuni pezzi su un unico pallet, sarà comunque perso per l'assemblaggio di un ulteriore pancale. Meno scontato appare invece il discorso relativo alla stabilità, dato che la pila che viene a formarsi dovrà necessariamente stare in piedi. Tuttavia come vedremo in seguito, è quasi sempre possibile rendere più stabile un pallet con l'ausilio di accorgimenti esterni come l'utilizzo di interfalde o di particolari colle adesive. Ovviamente questi

accorgimenti finiscono per incidere sui costi di produzione, e quindi sarebbe bene evitarli o quanto meno limitarne l'uso, e questo è appunto lo scopo della nostra ricerca. Da questo punto di vista risulta evidente che non possiamo limitarci all'uso di un algoritmo euristico ma dobbiamo avvalerci di un esatto. Oltretutto, volendo realizzare questo prodotto per scopi commerciali, non deve essere sottovalutata l'idea che l'eventuale utente possa sentirsi limitato dall'utilizzo di un algoritmo 'soltanto' euristico. La necessità di un algoritmo esatto appare quindi evidente. Vero è che in determinati casi un algoritmo euristico potrebbe rivelarsi decisamente più economico di uno esatto, ma altrettanto efficace. In simili situazioni l'euristico ci permetterebbe di evitare inutili analisi di layouts che risulterebbero comunque superflui. L'ideale sarebbe quindi di avere a disposizione due algoritmi di ricerca, uno esatto e uno euristico, e riuscire a stimare quando risulti più produttivo operare la ricerca dei layouts con uno piuttosto che con l'altro. Questa è la linea che abbiamo seguito nello sviluppo del nostro programma. A tal fine occorre stabilire un minimo di criteri mediante i quali poter stimare a priori quale algoritmo di ricerca sia meglio avviare.

Una prima discriminazione a favore dell'euristico si ha nel caso in cui i prodotti risultino con dimensioni l'una multipla dell'altra: è ovvio che un algoritmo esatto considererà tutte le possibili permutazioni dei pezzi sul pallet. Ciò produrrà un notevole numero di soluzioni che dovrebbero essere poi analizzate combinandole insieme per la stabilità e per il calcolo dell'efficienza dell'utilizzo della linea di trasmissione. Oltre tutto, dal punto di vista delle prese-rilasci, molte delle soluzioni trovate finirebbero soltanto per risultare un peggioramento della soluzione in cui i pezzi sono disposti nel modo più omogeneo possibile, mentre per quanto riguarda la stabilità spesso le cose cambiano di poco tra una soluzione e l'altra, quindi l'utilizzo di un algoritmo euristico potrebbe risultare preferibile.

Un'ulteriore discriminante potrebbe essere determinata stimando l'upper-buond del problema, ossia facendo una veloce valutazione del numero massimo di pezzi inseribili in uno strato: se l'upper buond risulta elevato, diciamo sopra i 50 pezzi per strato, allora l'utilizzo di un algoritmo esatto potrebbe risultare piuttosto lento e macchinoso e dovrebbe essere preferito l'utilizzo del più veloce

euristico. D'altro canto se il numero dei pezzi inseribili per layout è piuttosto contenuto e se le dimensioni dei pezzi non sono tra loro multiple, sarà sempre meglio tentare la strada dell'algoritmo esatto.

2.4. La funzione 'Genera Layers'

Seguendo questa linea, il programma PALLETTIZZA è stato dotato di una funzione denominata *GeneraLayers*, che produce i layouts degli schemi che inseriscono il maggior numero di prodotti per strato. Tale funzione riceve in ingresso le dimensioni del pallet e dei prodotti che può elaborare accedendo a due algoritmi di ricostruzione: uno pseudo-esatto e uno di tipo euristico a blocchi. La scelta dell'utilizzo dell'uno o dell'altro, in generale è affidata all'utente. Comunque all'interno del programma è stato inserito anche un valutatore delle dimensioni del prodotto e dell'upper bound del problema in modo che PALLETTIZZA, in assenza di scelte esterne, possa decidere in maniera autonoma quale algoritmo sia meglio utilizzare.

2.4.1. Definizioni

Nel seguito faremo riferimento ad una terminologia che è ormai diventata di uso comune nei PLP e viene quasi sempre adottata nella maggior parte degli studi inerenti. Sempre riferendoci al sistema di riferimento illustrato in Figura 2.5, in cui il pallet è orientato col lato lungo nella direzione 'Y' e col lato corto nella direzione 'X', si definisce *istanza* del problema la quaterna [L_pal , W_pal , L_box , W_box], in cui L_pal e W_pal rappresentano rispettivamente la lunghezza del lato lungo e del lato corto del pallet, e L_box , W_box rispettivamente la lunghezza del lato lungo e del lato corto della base d'appoggio del prodotto. Si definisce *partizione* lungo l'asse X una qualunque coppia di valori interi (n_x,m_x) tali che $X\_Pal \leq n_x*L\_box + m_x*W\_box$. Allo stesso modo una coppia di interi (n_y,m_y) risulta una partizione lungo l'asse Y se $L\_Pal \leq n_y*L\_box + m_y*W\_box$. I valori n,m della partizione assumono anche un significato fisico, in quanto finiranno per rappresentare il numero di pezzi orizzontali e verticali che potranno essere inseriti lungo una riga o una colonna del pancale, senza eccedere la sua lunghezza. Questo approccio monodimensionale è

spesso funzionale alla risoluzione del più generico problema di carattere bidimensionale in quanto si può cercare di massimizzare il numero di pezzi inseribili a partire da massimizzazioni perimetrali lungo i lati del pallet. In tale ottica risulterà più utile considerare le *partizioni efficienti*, ossia il sottoinsieme delle partizioni tali che non è inseribile alcun ulteriore pezzo lungo la linea. In termini matematici, ponendo b = min(L_box, W_box), possiamo definire partizioni efficienti lungo l'asse X quelle coppie (n_x,m_x) tali che $0 \leq W\_Pal - (n_x*L\_box + m_x*W\_box) < b$ e allo stesso modo saranno partizioni efficienti lungo l'asse Y quelle coppie (n_y,m_y) tali che $0 \leq L\_Pal - (n_y*L\_box + m_y*W\_box) < b$. Si definiranno infine *partizioni perfette* quelle coppie di valori, se esistono, tali che ricoprono perfettamente la linea, ossia lungo l'asse X dovrà risultare $W\_Pal = (n_x*L\_box + m_x*W\_box)$ mentre lungo l'asse Y si dovrà avere $L\_Pal = (n_y*L\_box + m_y*W\_box)$. E' chiaro che per ricoprire una superficie, l'ideale sarebbe utilizzare una serie di partizioni perfette o, in mancanza di queste, una serie di partizioni efficienti che meglio ricoprono la linea. A tal fine si può stimare l'efficacia di una partizione definendo il parametro *mancata copertura* come la quantità che manca alla partizione per completare la linea. Così per una partizione lungo l'asse X, la mancata copertura sarà data dalla differenza $W\_Pal - (n_x*L\_box + m_x*W\_box)$, mentre per una partizione lungo l'asse Y sarà $L\_Pal - (n_y*L\_box + m_y*W\_box)$. Ovviamente per una partizione perfetta la mancata copertura è nulla, mentre, in generale, per una qualunque partizione efficiente sarà sempre minore della quantità W_box.

2.4.2. Genera Layers euristico a blocchi

Prendendo spunto dagli euristici a 5 Blocchi, è stato implementato un algoritmo che considera tutte le possibili combinazioni di partizioni efficienti che possono verificarsi lungo i quattro lati del Pallet. Tale algoritmo è in grado di ricostruire tutti i layouts che abbiano le seguenti caratteristiche illustrate in Figura 2.7:

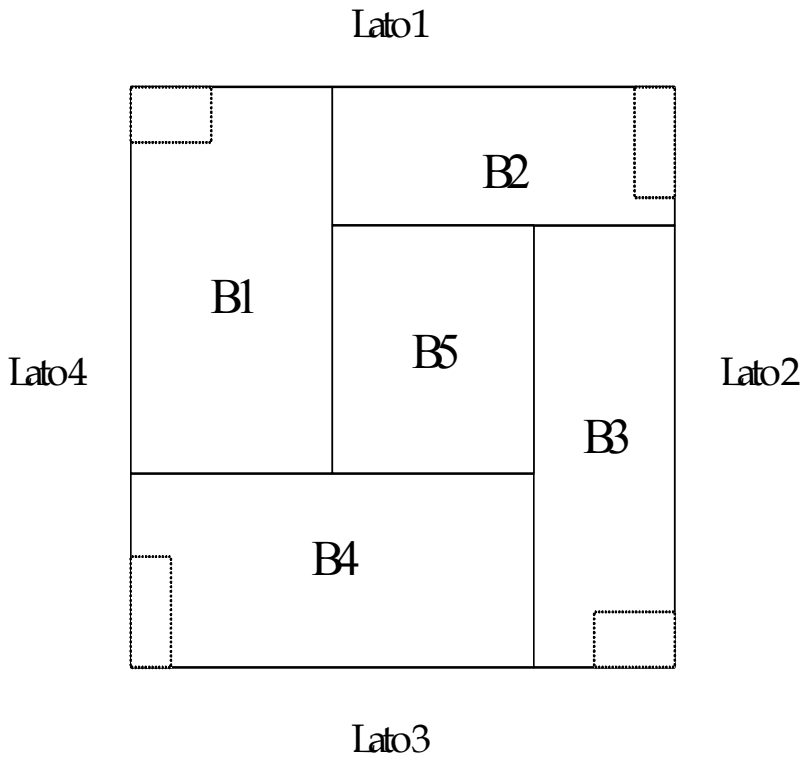


Figura 2.7: schema di ricostruzione dell'algoritmo euristico a blocchi

B1: Blocco omogeneo, con tutti i pezzi orizzontali, piazzati a partire dall'angolo alto sinistro

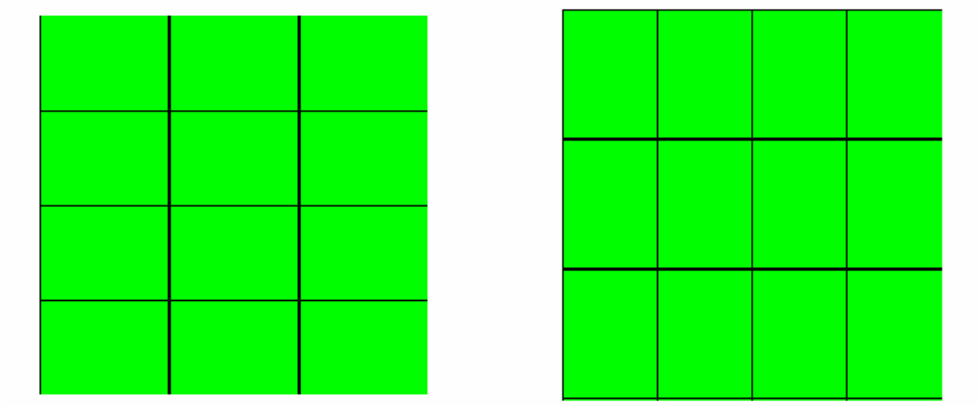
B2: Blocco omogeneo, con tutti i pezzi verticali, piazzati a partire dall'angolo alto destro

B3: Blocco omogeneo, con tutti i pezzi orizzontali, piazzati a partire dall'angolo basso destro

B4: Blocco omogeneo, con tutti i pezzi verticali, piazzati a partire dall'angolo basso sinistro

B5: Blocco omogeneo, con orientazione dei blocchi da definirsi.

Tali blocchi omogenei sono caratterizzati dall'avere tutti i pezzi con la stessa orientazione e tutti adiacenti l'uno all'altro. Quindi sono blocchi rettangolari e possono essere identificati dal numero dei pezzi per riga e per colonna e dalla loro orientazione. In questo modo possiamo classificare i blocchi come Blocco-Orientazione-[numero_ righe x numero_ colonne]:



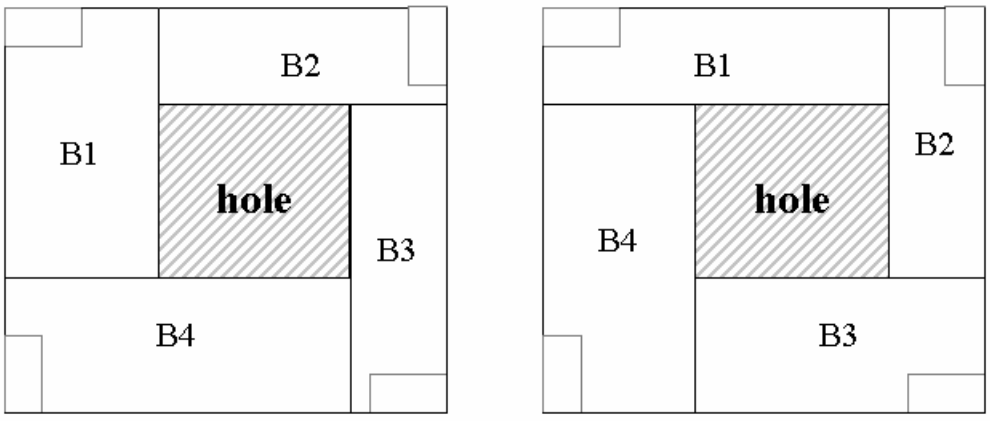
Blocco H [4x3] Blocco V [3x4]

Figura 2.8: schematizzazione dei blocchi possibili

Note tutte le partizioni efficienti lungo i quattro lati del pallet è banale ricostruire i blocchi d'angolo. Se (n_i,m_i) è la partizione sullo *i*-esimo lato, si ha subito: B1 = Blocco H [n_4 x n_1], B2 = Blocco V [m_1 x m_2], B3 = Blocco H [n_2 x n_3], B4 = Blocco V [m_4 x m_3]. La ricostruzione del layout è possibile nel caso in cui non ci siano sovrapposizioni dei vari blocchi. Una volta verificato che ciò non accade si passa alla valutazione dell'eventuale buco centrale ed eventualmente, se è sufficientemente grande, al suo riempimento.

Per la sua valutazione, si possono fare le seguenti considerazioni di carattere pratico:

Possiamo distinguere due tipologie diverse di buco centrale, a seconda della sua collocazione rispetto ai blocchi perimetrali, come illustrato in Figura 2.9:



buco sotto B2 e a destra di B1 buco sotto B1 e a destra di B4

Figura 2.9: schemi a blocchi, identificazione del buco centrale

Per la valutazione delle dimensioni e la collocazione del buco centrale basterà quindi stabilire da quale di queste due configurazioni esso derivi.

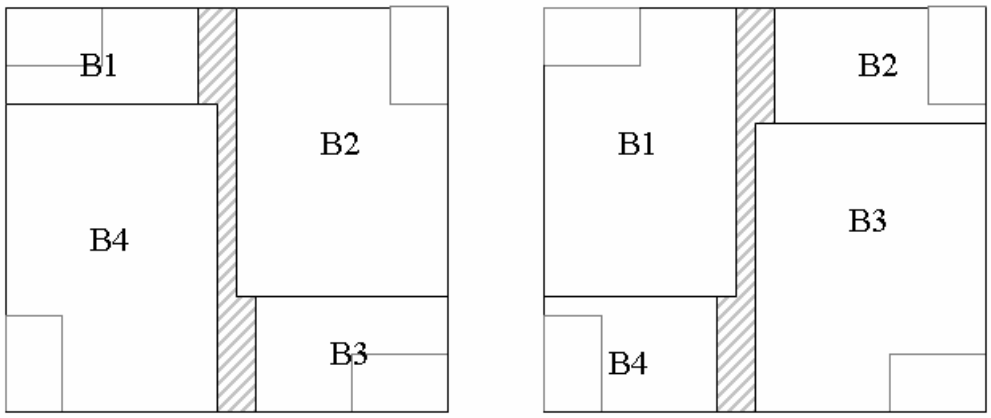


Figura 2.10: schemi a blocchi, assenza del buco centrale

Si tenga inoltre presente che situazioni del tipo illustrate in Figura 2.10, possono verificarsi nella pratica, ma la parte tratteggiata non deve essere considerata un buco, in quanto non è riempibile. Infatti la parte più ampia è ai bordi del pallet e qui, per la definizione stessa di partizione efficiente, non potrà essere inserito alcun pezzo ulteriore.

Un altro caso in cui non ha alcun senso considerare un eventuale buco è quando viene a mancare almeno uno dei quattro blocchi perimetrali come illustrato in Figura 2.11. Le configurazioni a 5 blocchi sono caratterizzate da 4 blocchi d'angolo e da uno centrale. Venendo a mancare un blocco d'angolo non ha senso cercare la configurazione con il blocco centrale. Questo si può verificare quando le combinazione delle partizioni efficienti sui quattro lati generano uno o più blocchi di area nulla.

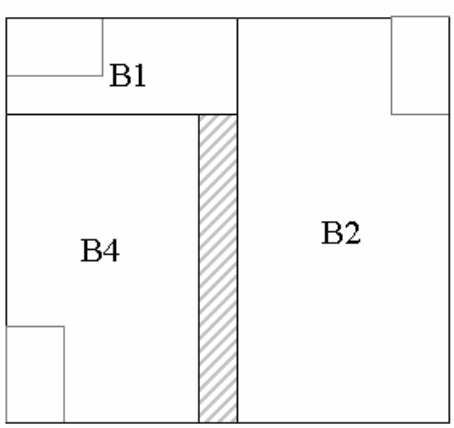


Figura 2.11: degenerazione in uno schema a 3 blocchi

Tenendo conto di queste considerazioni è facile implementare un algoritmo che faccia tutte le possibili combinazioni di partizioni efficienti e generi i layouts che inseriscono più pezzi. Dal punto di vista computazionale un simile algoritmo è piuttosto semplice e quindi estremamente veloce. Essendo costituito da blocchi omogenei il layout che ne deriva risulterà anche particolarmente comodo dal punto di vista delle prese-rilasci.

2.4.3. Genera Layers pseudo-esatto

Partendo dalle considerazioni di Bhattacharia [15] si è cercato di realizzare un algoritmo che data una certa istanza, sia in grado di fornire layouts diversi a meno di eventuali traslazioni.

Sempre rifacendosi al sistema di riferimento di Figura 2.5, l'idea dell'algoritmo è quella di partire da un angolo del pallet, nel nostro caso dall'angolo posto in (0,0), e quindi cercare di inserire più pezzi possibile lungo il lato orizzontale, successivamente si sale di ordinata e si cerca sempre nuovamente di riempire il pallet lungo la linea orizzontale. Riempire al massimo una linea significa considerare una sua partizione efficiente. Di conseguenza una volta determinate tutte le possibili partizioni efficienti lungo l'asse X, si devono analizzare tutte le loro combinazioni per valutare quali siano le configurazioni che inseriscono il numero massimo di prodotti per layout. A tal fine si può generare un algoritmo ad albero considerando i seguenti parametri che identificano i vari nodi dell'albero:

S_P: *Starting length*, è l'ordinata a cui inizia il nodo

dl_p: *Decision length*, sarà l'ordinata di inizio del nodo successivo dell'algoritmo.

F_p: *Fixed pieces*, è il numero di pezzi fissati ad una determinata dl_p

A_P: *Set of partially placed pieces*, è l'insieme dei pezzi parzialmente piazzati nel senso che la loro collocazione non è stata ancora assegnata: la loro assegnazione sarà confermata o meno dall'evolversi dell'algoritmo nei nodi successivi. Si noti che tale insieme tiene anche conto dell'orientazione dei pezzi parzialmente piazzati.

L'algoritmo si ferma quando la dl_p non permette alcun ulteriore inserimento di pezzi.

L'implementazione prevede una strategia di tipo DFS (depth frist search strategy) che ben si presta all'inserimento di opportune regole di taglio che sono necessarie per velocizzare la risoluzione nei casi non banali. Infatti implementando una strategia DFS ogni nodo è sviscerato fino al suo ramo più estremo: in questo modo prima di iniziare l'analisi del nodo in questione, si può sempre stimare il massimo numero di pezzi da lì inseribili. Nel caso che tale stima sia inferiore ai risultati già ottenuti, sarà inutile scandagliare il nodo che verrà quindi tagliato dall'analisi. Inoltre applicando una simile regola di taglio è chiaro che, tanto prima si trovi la soluzione ottima, tanto più tale regola sarà sfruttata evitando di andare ad analizzare casi non significativi. A tal scopo le partizioni efficienti vengono riordinate in base alla mancata copertura della linea. Sarà infatti molto più probabile che il layout che inserisce il numero massimo di pezzi sia dato da una serie di partizioni efficienti a mancata copertura più bassa delle altre. Con questi accorgimenti l'algoritmo funziona in tempi più che accettabili mentre omettendoli, i tempi computazionali crescono a dismisura con il rischio di impiegare anche qualche ora per l'analisi di istanze di media complessità. Da quanto detto è subito evidente che l'implementazione dell'algoritmo basata su una strategia di ricerca di tipo BFS (breadth frist search), non avrebbe potuto avvalersi di simili regole e quindi non sarebbe risultata altrettanto efficiente.

Realizzando un algoritmo con questo metodo alla fine otterremo una lista di nodi caratterizzanti i layouts che inseriscono il numero massimo di pezzi. Questa lista ci permetterà di ricavare

facilmente l'ordinata e l'orientazione dei pezzi inseriti, in quanto l'ordinata coinciderà con la S_p a cui è avvenuto l'inserimento del pezzo: la descrizione del pezzo inserito è nell'insieme A_p, da cui si ricava quindi anche l'orientazione. A questo punto il problema di fondo è la determinazione dell'ascissa. Potremmo applicare la teoria dei grafi rifacendoci all'idea di K.A.Dowsland e considerare un grafo i cui vertici rappresentino l'insieme di tutte le possibili posizioni dei pezzi, e che risultino adiacenti se le superfici occupate dai rispettivi pezzi si sovrappongono. Trovando poi i sottoinsiemi dei vertici del grafo associato tali che non ve ne siano di adiacenti, avremmo risolto il problema. Naturalmente qui avremmo molti vincoli in più rispetto a quanto fatto dalla stessa K.A.Dowsland in [14], in quanto conosceremmo già l'ordinata dei pezzi e quindi l'algoritmo dovrebbe essere notevolmente semplificato rispetto a quello descritto nel citato articolo. Tuttavia in generale, rimarrebbe la questione di trovare più layouts associati ad una stessa lista, tutti molto simili tra loro. Per limitare le soluzioni sarebbe invece preferibile associare subito un unico layout ad ogni lista ed eventualmente aggiustare in seguito la sua configurazione in base alle esigenze. Innanzi tutto diciamo che relativamente al problema delle prese, sarebbe conveniente che tutti i pezzi con la stessa orientazione che partono dalla stessa ordinata, fossero adiacenti. A tal fine può risultare piuttosto interessante la lettura dell'articolo [16], nel quale viene descritto un metodo per lo stoccaggio di container, in cui devono essere inseriti oggetti di tipo diverso anche se sempre con forma a parallelepipedo. Il problema è al solito ricondotto al caso 2D considerando il container come formato dalla sovrapposizione di più strati. Venendo a mancare il vincolo del monoprodotto però, per la realizzazione dell'algoritmo che sia in grado di generare gli strati utili, le cose si complicano. L'approccio è di tipo euristico: ad ogni inserimento si cerca di minimizzare l'area che risulterà inevitabilmente persa per gli inserimenti successivi. A tal fine si procede al riempimento dello strato, cercando di lasciare più regolari possibili le superfici libere restanti. Ad esempio, nel caso rappresentato in Figura 2.12

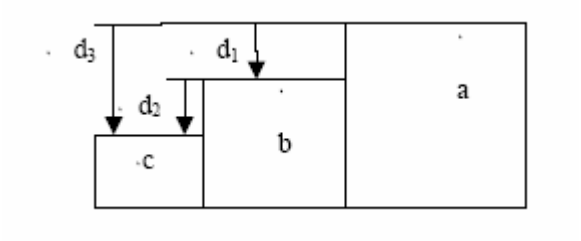


Figura 2.12: riempimento di un container con prodotti di forma diversa

si dovrà, per prima cosa procedere al riempimento della superficie sopra il blocco ‘c’ di ampiezza d2, in modo da ripristinare un’area di forma rettangolare sopra i blocchi ‘c’ e ‘b’. Successivamente si cercherà di riempire tale area per ripristinare la zona a livello d3. A meno che non ci sia altra soluzione, non avrebbe molto senso inserire a priori un blocco alto più di d3 sopra ‘c’, in quanto finirebbe soltanto per rendere più difficoltosa la copertura dell’area sopra ‘b’ di ampiezza d1. Con molta probabilità questa superficie diverrebbe area persa per i successivi inserimenti.

Nel nostro caso monoprodotto, queste regole si traducono in un inserimento omogeneo dei pezzi che partono dalla stessa ordinata. In generale quindi se abbiamo più pezzi con uguale orientazione che partono tutti dalla stessa ordinata, sarà quasi sempre conveniente inserirli uno di seguito all’altro. Questo risultato come detto sarà molto comodo anche dal punto di vista dell’analisi delle prese-rilasci, e dunque non ci sono particolari ragioni di andare alla ricerca di layouts più strani che magari sparpaglino i pezzi lungo la linea: ciò finirebbe solo per appesantire l’algoritmo di ricerca che forse riuscirebbe anche a determinare ulteriori strati, ma che con molta probabilità sarebbero scartati o quantomeno post preferiti, dalle successive analisi delle prese e della stabilità. Di conseguenza l’algoritmo di ricostruzione cercherà di piazzare i pezzi omogenei che partono dalla stessa linea, l’uno adiacente all’altro, sempre cercando di minimizzare l’area persa. Il problema della minimizzazione dell’area persa può essere risolto in maniera euristica considerando i vari casi qui di seguito illustrati:

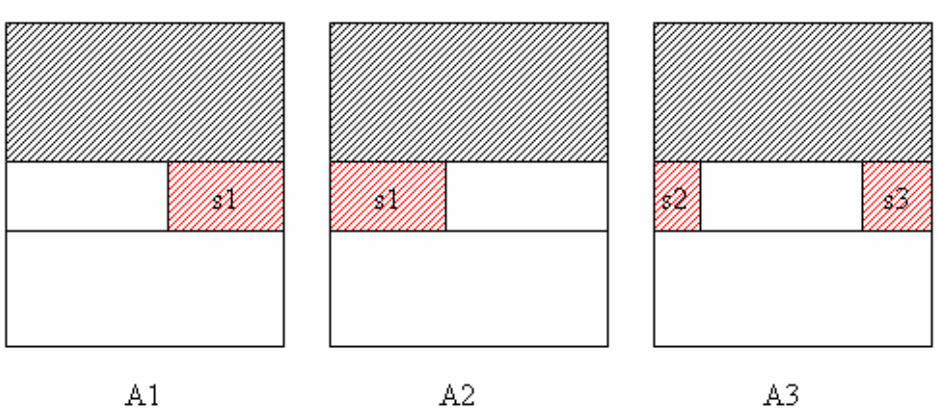


Figura 2.13: riempimento del pallet lungo una linea libera

La presenza di tutta la linea libera è ovviamente la più favorevole e sarà sufficiente allineare il blocco lungo un lato esterno del pallet come in A1 o A2. Il piazzamento del blocco come in A3 è invece da sconsigliare, in quanto nonostante la somma delle aree riutilizzabili s2+s3 sia uguale all'area s1, le due aree proprio in quanto più piccole, prese singolarmente, sarebbero di più difficile riutilizzo per il seguito della pallettizzazione. Il piazzamento come in A3 risulterebbe corretto solo a patto che i pezzi del blocco coprano un'intera partizione efficiente del pallet: in tal caso, per la definizione stessa di partizione efficiente, tutta la superficie s1 sarebbe comunque area persa non più utilizzabile, e quindi potrebbe essere ridistribuita in qualunque modo lungo la striscia.

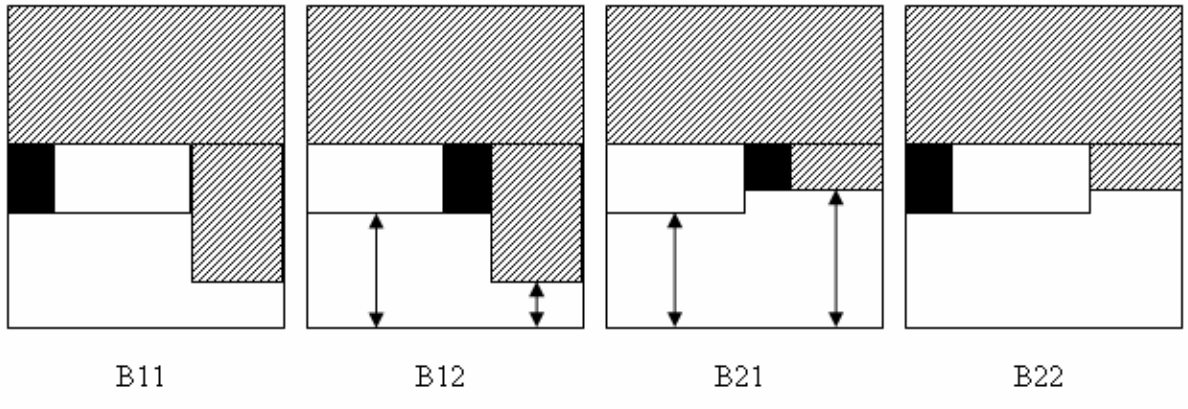


Figura 2.14: riempimento di una linea del pallet occupata su un lato

Nel caso invece che un lato del pallet sia già occupato da altri pezzi, per l'allineamento del nuovo blocco, dovrebbe essere tenuta in considerazione l'altezza dell'area già occupata. Le figure B11 e

B12 mostrano il caso in cui l'area occupata scenda sotto un livello più basso del blocco da inserire: in simili circostanze l'allineamento del blocco lungo il lato libero del pallet (vedi B11), o a ridosso dell'area occupata (vedi B12), sono equivalenti dal punto di vista dell'area persa. Al contrario quando è il blocco ad avere ampiezza maggiore rispetto all'area adiacente già occupata, il suo inserimento va sempre fatto dalla parte del lato libero del pallet come in figura B21, pena l'aumento di area persa, vedi B22. In generale si può concludere che se c'è spazio sufficiente per allineare un blocco lungo un lato esterno del pallet, tale inserimento è sempre corretto. Si osservi che, riordinando e inserendo i pezzi per ordinate crescenti, le situazioni di figura B11 e B22 non si potranno mai verificare per i blocchi formati da pezzi verticali: infatti tutti i pezzi precedentemente piazzati avrebbero al più ampiezza uguale ma sarebbero stati inseriti a ordinate minori.

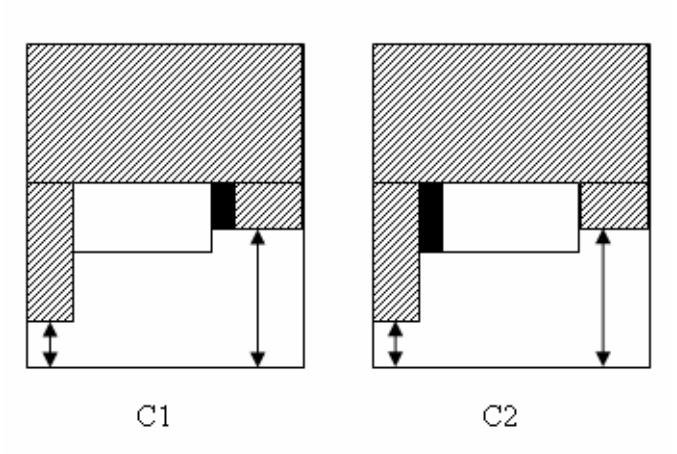


Figura 2.15: riempimento di una linea del pallet occupata su ambo i lati

Infine l'inserimento di un blocco che si trovi entrambi i lati del pallet occupati andrà sempre preferito a ridosso dell'area occupata di ampiezza maggiore, come illustrato in figura C1. In questa situazione valutando l'eventuale area persa si può avere una stima della bontà o meno dell'inserimento in esame.

Tutte queste considerazioni vanno bene nel caso che da una certa ordinata si stacchino tutti e soli pezzi con la stessa orientazione. La casistica è invece assai più delicata nel caso in cui si debbano inserire sia pezzi con orientazione verticale che con orientazione orizzontale. Per quanto già detto non avrebbe alcun senso cercare di inserire i pezzi in maniera alternata V-H-V-H-V-H…, ma

converrà applicare un inserimento a blocchi collocando prima i pezzi di un tipo e poi i pezzi dell'altro. Tuttavia l'inserimento prima degli uni e poi degli altri o viceversa non è affatto indifferente: iniziare l'inserimento con un tipo piuttosto che con l'altro può condurre a situazione in cui, da un certo punto in poi, venga a mancare lo spazio sufficiente per poter ricostruire il layout fino in fondo. Di conseguenza quando da una certa ordinata si staccano pezzi di tipo diverso, si dovranno valutare tutte le possibilità di inserzione, considerando prima il collocamento dei pezzi orizzontali e poi quello dei verticali o viceversa. Possiamo quantificare il numero di combinazioni da dover analizzare: poiché si hanno soltanto due tipi diversi di orientazione, se N è il numero di ordinate da cui partono entrambi i tipi di pezzi, 2^N sarà il numero di combinazioni totali da prendere in considerazione. Non sarà necessario valutarle tutte, ma basterà fermarci non appena si trova una combinazione che ricostruisce il layout completo. Questa situazione può verificarsi quando lungo la prima linea del pallet vengono inseriti sia pezzi orizzontali che verticali: in questo caso secondo il sistema di riferimento adottato, i pezzi inseriti staccano tutti dalla medesima ordinata nulla. Tale situazione è comunque stata presa in considerazione dallo stesso Bhattacharia [15], il quale afferma che, tra tutte le soluzioni possibili di disporre il numero massimo di pezzi, sicuramente ce ne è una che vede collocati sulla prima linea del pallet prima i pezzi verticali e poi quelli orizzontali. Questa è appunto la regola che verrà applicata per la ricostruzione della prima linea del nostro layout. Il risultato cui è giunto Bhattacharia, non è affatto in contraddizione con le osservazioni fatte a proposito dell'area persa: si consideri a tal fine la Figura 2.16: inserendo prima i pezzi verticali e poi i pezzi orizzontali, come in D1, si minimizza sempre l'area persa; non sarebbe affatto la stessa cosa fare il contrario, ossia inserire prima i pezzi orizzontali e poi i pezzi verticali, vedi D2, in quanto l'area persa non sarebbe più la stessa. Come proposto dall'algoritmo elaborato per la ricostruzione, potremmo pensare di inserire prima gli orizzontali e poi i verticali allineando questi ultimi sul lato esterno del pallet, vedi D3. Tuttavia con questo inserimento finiremmo solo per trovare delle soluzioni simmetriche rispetto al caso precedente. L'idea di Bhattacharia ci permette quindi di evitare di considerare soluzioni doppioni nel caso in cui lungo la prima linea del pallet si

verifichi l'inserimento di entrambi i tipi di pezzi. In generale questa regola potrà essere applicata in ogni situazione in cui si debbano inserire sia pezzi orizzontali che verticali lungo una qualsiasi linea del pallet, non necessariamente la prima, che comunque risulti completamente libera: in tal caso andrà sempre bene prima l'inserimento dei verticali e poi degli orizzontali.

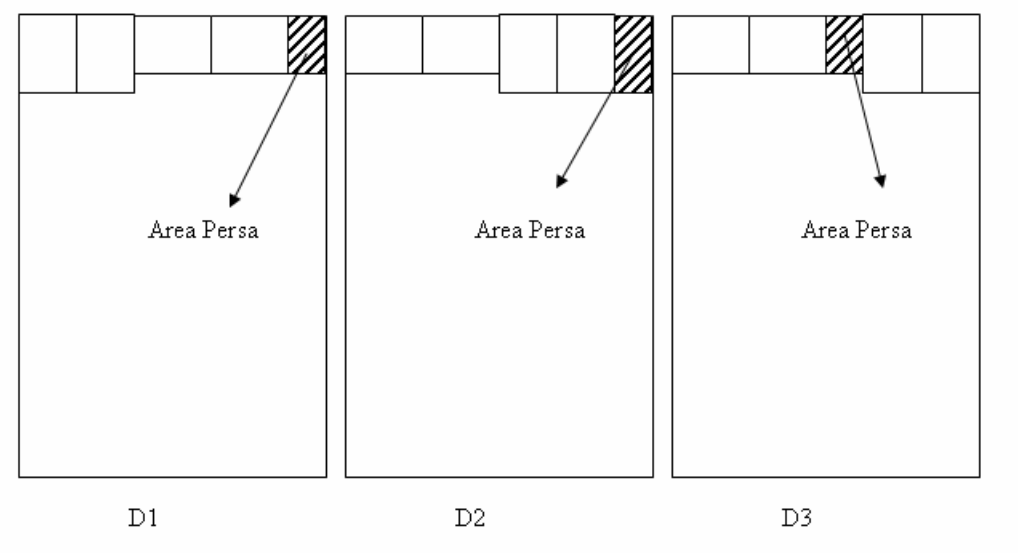


Figura 2.16: inserimento dei prodotti lungo la prima linea del pallet

I problemi dovuti all'inserimento di pezzi sia orizzontali che verticali lungo una linea non è libera, è evidenziato in Figura 2.17:

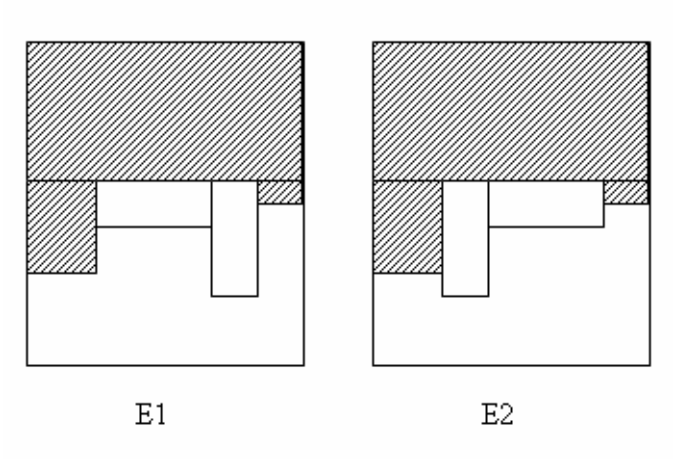


Figura 2.17: inserimento di prodotti H, V lungo una linea già occupata su entrambi i lati

In simili casi l'area che rimane per i successivi inserimenti, viene quasi sempre suddivisa in diverse regioni. Valutare a priori, quale sia il miglior modo di ripartirla, è piuttosto difficile in quanto

dipende dagli stessi inserimenti successivi. In simili casi sarà necessario valutare entrambe le possibilità.

Fatte tutte queste considerazioni, possiamo ricavare le seguenti regole di natura pratica-intuitiva che permettono di elaborare l'algoritmo di ricostruzione illustrato in Figura 2.18:

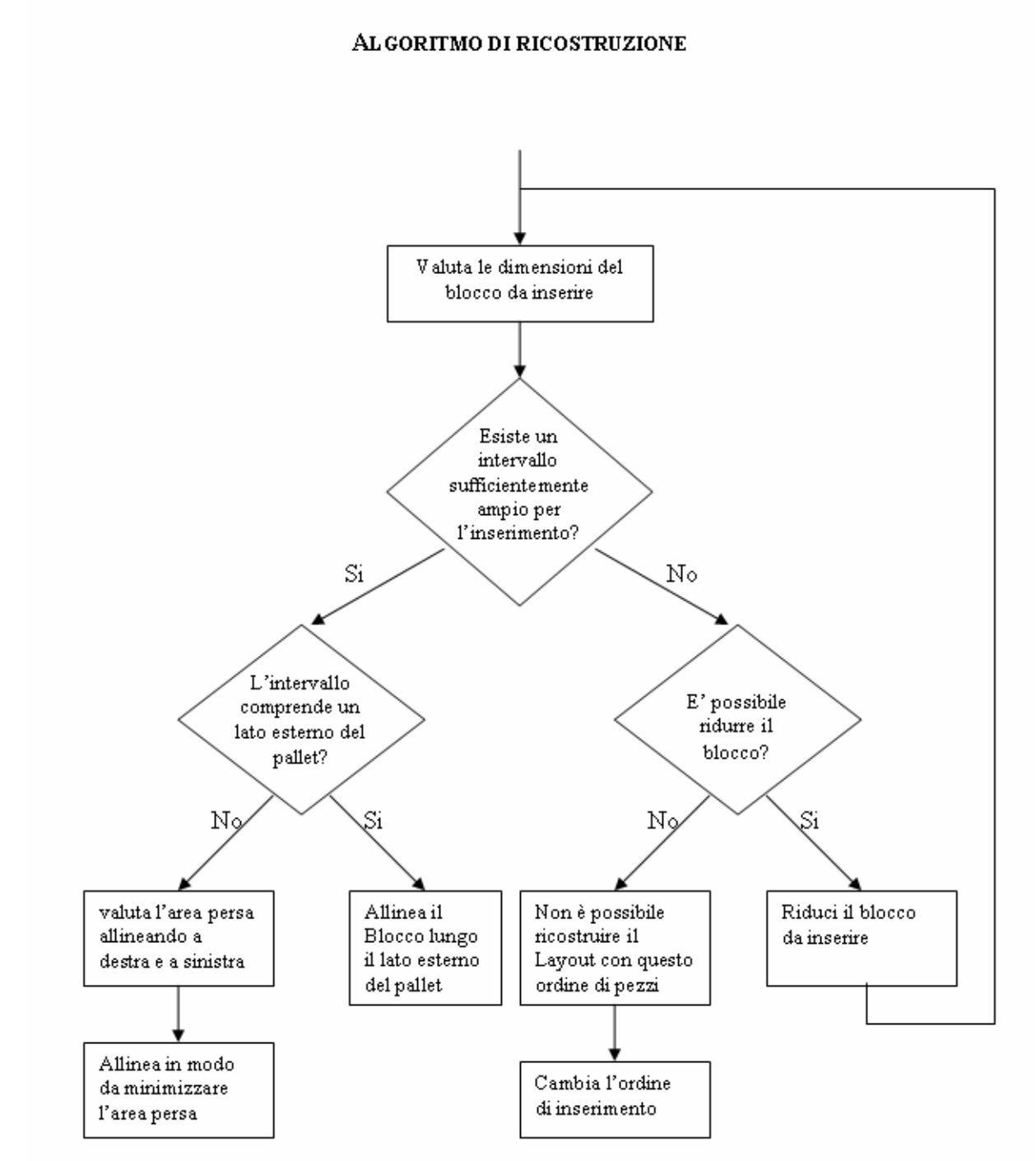


Figura 2.18: ***Algoritmo di ricostruzione dei layers***

Si osservi che l'algoritmo di ricostruzione associa uno ed un solo layout ad ogni soluzione proposta dall'algoritmo di Bhattacharia. Ad essa però potrebbero essere associati più layout diversi, ottenuti magari per traslazione di alcuni pezzi o per varie permutazioni. Le permutazioni non ci interessano più di tanto, perché con molta probabilità finirebbero soltanto per produrre layout con caratteristiche molto simili tra loro. Al contrario le traslazioni, specie nel caso che separino pezzi adiacenti con la stessa orientazione, possono influire direttamente sui parametri di stabilità e di efficienza delle prese. In tale ottica è stata implementata una funzione, denominata *Perimetrizza*, che ha il compito di rifinire il layout dopo la ricostruzione: essa identifica e definisce eventuali traslazioni di pezzi lungo i bordi esterni del pallet. Come vedremo dall'analisi della stabilità, le zone esterne sono particolarmente delicate ed è sempre bene cercare di riempirle. Traslare i prodotti distanziandoli leggermente l'uno dall'altro, produrrà un incremento dei rilasci necessari alla ricostruzione, ma anche la frammentazione dello spazio di traslazione in buchi più piccoli che saranno meno insidiosi per la stabilità finale. Al contrario traslare i prodotti mantenendoli uniti, lascerà inalterata l'efficienza delle prese, ma traslerà anche l'intero buco in una zona più interna del pallet: qui sarà certamente meno pericoloso che a ridosso dei bordi esterni, ma potrebbe essere ugualmente dannoso per la stabilità. La funzione *Perimetrizza* cercherà di rifinire il layout più orientato alla stabilità o all'efficienza delle prese in base alle esigenze dell'utente.

2.4.4. Valutazione Upper Bound

La prima idea che viene in mente per la valutazione dell'upper bound è ovviamente quella di dividere l'area pallettizzabile per l'area della superficie d'appoggio dei prodotti. Il risultato preso per difetto risulterà certamente un upper bound del nostro problema.

$$N_{UB} = \frac{S}{A} \qquad (1)$$

S = Superficie da ricoprire

A = Area d'appoggio del prodotto

Il problema di questa semplice espressione è che se consideriamo S come l'intera superficie del pallet, la stima che si ottiene può risultare piuttosto grossolana e rischia di non essere d'aiuto in molti casi pratici. Il fatto è che non si tiene conto dell'*area inutilizzabile*, ossia della superficie del pallet che non potrà mai essere ricoperta da un numero intero di pezzi delle dimensioni date. Se questa superficie è maggiore dell'area d'appoggio di un singolo pezzo, la stima dell'upper bound fornita dalla (1) risulterà errata per eccesso. Si può comunque cercare di migliorare la stima seguendo le idee di F.W.Barnes [17], e andare a valutare l'area inutilizzabile per poi sottrarla alla superficie complessiva del pallet. Per la stima dell'area inutilizzabile ancora una volta risultano utili i concetti inerenti alle partizioni: qualunque combinazione di pezzi lungo una linea, non potrà mai ricoprirla più della partizione di lunghezza massima.

Siano quindi i il numero di partizioni lungo un certo lato del pallet, ed L lunghezza massima delle partizioni, risulterà:

$$L = \bar{n_i} \cdot L_{box} + \bar{m_i} \cdot W_{box} \qquad \text{se} \qquad \text{per } \forall i \qquad L \geq n_i \cdot L_{box} + m_i \cdot W_{box}$$

E' chiaro che se è presente una partizione perfetta, L coinciderà con la lunghezza effettiva del pallet. Il procedimento può essere ripetuto lungo il lato X ed Y del pancale, ottenendo una L_x, e L_y.

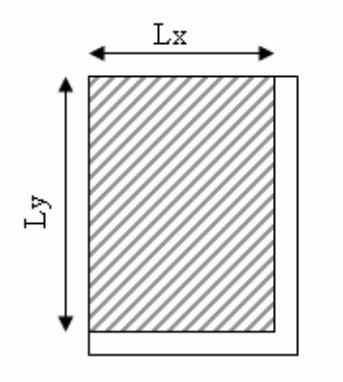


Figura 2.19: migliore approssimazione dell'area pallettizzabile

In questo modo, la superficie che effettivamente può essere ricoperta sarà data dall'area del rettangolo tratteggiato in figura .

$$S = L_x \cdot L_y \qquad (2)$$

Andando a sostituire questo risultato nel computo dell'espressione (1), otterremo una buona e veloce stima dell'upper bound del problema. Si osservi che questa soluzione è ancora una stima per eccesso, e non assicura che all'interno del pallet siano inseribili esattamente N_{UB} prodotti per strato. In generale risulterà:

$$N_{prodotti\ inseribili\ per\ strato} \leq N_{UB} \qquad (3)$$

Sul problema dell'upper bound sono stati effettuati diversi studi e alcuni autori [18] [19] hanno anche cercato una sua determinazione esatta e non soltanto approssimata, elaborando dei veri e propri algoritmi. Questo perché la conoscenza a priori del numero massimo di prodotti inseribili per strato può fornire informazioni utili sulla bontà o meno degli algoritmi utilizzati, o può condurre ad una ricerca mirata negli algoritmi di tipo esatto con conseguente recupero del tempo perso per il calcolo del limite dei pezzi inseribili. Ad esempio, come visto nell'algoritmo di Bhattacharia, l'utilizzo di un upper bound può essere utile per l'introduzione di utili regole di taglio che evitano l'esplosione dei calcoli computazionali. In questo modo, molte delle idee alla base del calcolo dell'upper bound finiscono per essere efficacemente sfruttate negli stessi algoritmi di ricostruzione dei layouts.

Capitolo 3 : Ottimizzazione dell'utilizzo del sistema di automazione

Fino ad ora abbiamo considerato i vari strati che dovranno andare a comporre il pallet completo come l'insieme di più pezzi, tutti di forma uguale, eventualmente con diversa orientazione. Questa visione dello strato composto dai singoli prodotti è risultata utile per la sua ricostruzione e lo sarà anche dopo per l'analisi della stabilità; tuttavia dal punto di vista delle prese-rilasci risulta più comodo allontanarsi da questa idea dello strato formato da singoli prodotti, e passare ad una visione in formato prese-rilasci, ossia è più adeguato pensare lo strato come formato dai blocchi di pezzi che formano le varie prese-rilasci. In fondo questo è il modo in cui il sistema automatizzato vede e ricostruisce il layout dello strato. Cominciamo quindi col dire che una presa-rilascio sarà costituita da un numero massimo di pezzi, tutti adiacenti tra loro, e con la stessa orientazione. In questo caso l'end-effector del robot utilizzato per la pallettizzazione potrà prelevarli tutti insieme dal fine linea e depositarli sul pallet con un unico movimento. Ovviamente il medesimo strato potrà essere ricostruito in modi diversi, utilizzando prese-rilasci diverse. Si dovrà quindi cercare di realizzare un algoritmo che, partendo dal layout dello strato in visione di singoli pezzi e dato un certo sistema di automazione, stabilisca qual è il modo migliore di ricomporlo fisicamente sul pallet. Nel valutare l'ottimizzazione di questo processo si dovranno tenere in considerazione i seguenti fattori:

1. Numero di prese necessarie alla pallettizzazione
2. Utilizzo della linea di trasmissione
3. Ordinamento delle prese

Per quanto riguarda le prese necessarie è evidente che occorre cercare di minimizzarne il numero, ottenendo così la stessa pallettizzazione in minor tempo. Questo sarà il problema prioritario nell'analisi delle prese in quanto, tra due versioni dello stesso strato, una realizzata a prese minime e

l'altra no, verrà sempre preferita quella a prese minime. Tuttavia è facile rendersi conto che in generale, una stessa configurazione può essere ricostruita con più soluzione diverse, tutte a prese minime, ossia non è detto che vi sia una corrispondenza biunivoca tra un layout in versione pezzi e lo stesso layout in versione prese: solitamente allo stesso layout in versione pezzi potranno essere associati più layouts diversi in versione prese, tutti comunque a prese minime. Generalmente ricostruire lo strato utilizzando un set di prese minime piuttosto che un altro, non sarà la stessa cosa. A tal riguardo bisogna tener presente che i prodotti arrivano in un certo tempo al sistema automatizzato. Questi prodotti vengono accumulati al fine linea. E' chiaro che se il robot adibito alla pallettizzazione può prendere più prodotti contemporaneamente, ma questi non sono stati ancora tutti accumulati, si dovrà aspettare che l'accumulo sia completo. In questo modo si perde del tempo in cui il robot sta fermo e non fa niente. Per ovviare a questo inconveniente si potrebbe pensare di gestire opportunamente la velocità di accumulo al fine linea in base alle esigenze delsistema, ma è anche vero che questa sofisticazione non sarebbe necessaria nel caso che l'utilizzo dei pezzi al fine linea fosse pressoché costante. Questa è quindi la soluzione da preferire.

Pertanto, nell'analisi dell'utilizzo della linea di trasmissione, si dovrà valutare l'omogeneità delle prese. Tale analisi servirà a scegliere tra due o più versioni, tutte a prese minime, quella che utilizza meglio la linea di trasmissione.

Una volta stabilite le prese-rilasci da effettuare, occorre stabilirne l'ordinamento. Infatti durante l'inserimento dei prodotti sul pallet, il robot deve avere sempre spazio sufficiente per manovrare. Ciò significa che non può essere fatto un ordinamento casuale, ma andrà opportunamente valutato in base alle esigenze del robot utilizzato. L'accosto dei prodotti gli uni agli altri, è infatti una procedura piuttosto delicata che deve essere eseguita con maggiore accuratezza: in particolare sono sempre necessari almeno due lati liberi nell'area di inserimento che consentano il movimento della pinza. In base a questa proprietà andrà riordinata la sequenza di prese rilasci.

3.1. Numero di prese necessaire

Innanzi tutto diciamo che di norma, uno strato di nostro interesse sarà costituito da 20-30 pezzi, 50 pezzi al massimo. Considerando che i robot attuali riescono ad afferrare anche sei pezzi per volta, solitamente lo strato sarà ricostruito con meno di dieci prese: ciò significa che anche impiegare una presa in più rispetto alla soluzione ottima porterebbe a compiere un errore del 10%. Di conseguenza sarebbe importante realizzare un algoritmo esatto in grado di stimare in maniera rigorosa il numero minimo di prese necessario per la pallettizzazione. Al solito però anche questo è un problema di tipo NP. Tanto per capirne la complessità, un'idea per stimare il numero di prese potrebbe essere quella di ricondursi al problema della pallettizzazione, cercando di riempire lo spazio a disposizione non con il numero massimo di box rettangolari tutti uguali tra loro, bensì col numero minore di box, sempre a forma rettangolare, ma con dimensioni variabili e definite dal tipo di prese possibili [20]. L'alternativa che è la strada che abbiamo seguito, è quella di realizzare un algoritmo basato sulla teoria dei grafi. L'idea è suggerita da K.A.Dowsland sempre in [20]: in pratica ad ogni layout viene associato un grafo i cui nodi rappresentano i prodotti del layout, e i vari collegamenti tra nodi identificano la contiguità o meno tra i prodotti stessi, ovvero identificano la possibilità di poterli prendere insieme con un unico movimento del robot, oppure no.

A tal fine, l'algoritmo implementato compie le seguenti analisi:

1. il layout è scomposto in sottoblocchi omogenei
2. per ogni sottoblocco omogeneo vengono calcolate tutte le possibili prese e valutate tutte le loro possibili combinazioni
3. viene scelta la soluzione migliore considerando in sequenza i seguenti criteri:
    a. numero minimo di prese
    b. utilizzo della linea di trasmissione

3.1.1. Scomposizione in sottoblocchi omogenei

Definiamo un *sottoblocco omogeneo* l'insieme dei prodotti di uno strato tali che risultino tutti con la stessa orientazione e adiacenti l'uno all'altro. L'identificazione di tali sottoblocchi è di fondamentale importanza per la risoluzione del problema in quanto, due o più prodotti potranno essere afferrati insieme dall'end effector del robot solo se appartengono ad uno stesso sottoblocco omogeneo. Una volta scomposto il layout dello strato in tutti i suoi sottoblocchi omogenei che lo compongono, basterà valutare come ricostruire dal punto di vista delle prese, ogni singolo sottoblocco.

Esempio: consideriamo una delle soluzioni proposte da GeneraLayers per l'istanza [800,660,200,120], rappresentata nella Figura 3.1:

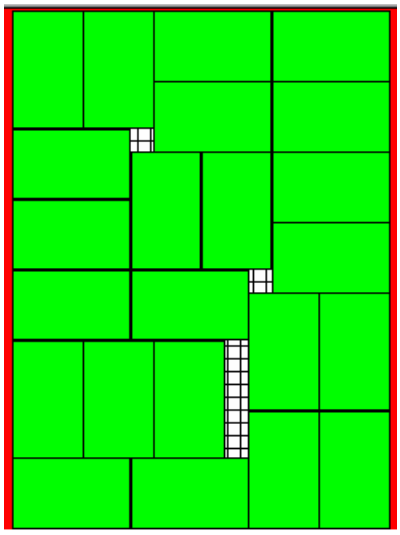


Figura 3.1: un layout possibile per l'istanza [800,660,200,120]

Tale layout viene passato come parametro ad una funzione che lo scompone nei suoi sottoblocchi omogenei illustrati in Figura 3.2:

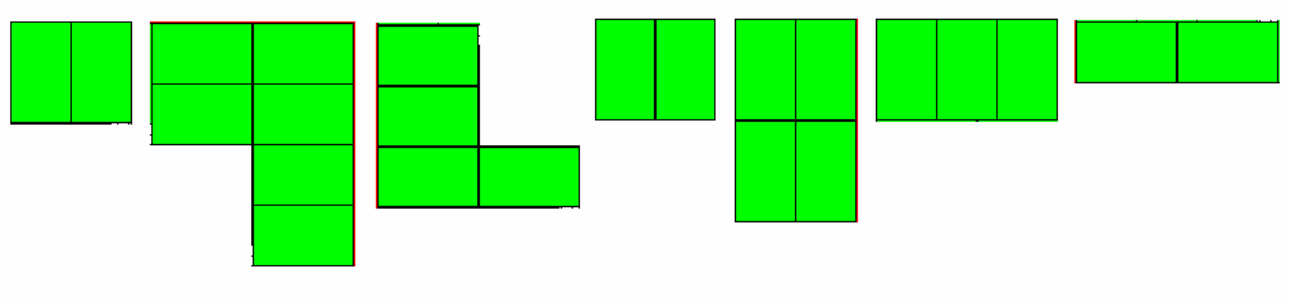


Figura 3.2: scomposizione in sottoblocchi per il layout di figura 1

A questo punto siamo in grado di analizzare ognuno di questi sottoblocchi dal punto di vista delle prese.

3.1.2. Determinazione delle prese per un sottoblocco

L'idea è quella di sviluppare un algoritmo ad albero che partendo da una lista di tutte le prese possibili, le combini insieme per valutare la soluzione che ne impiega meno. L'analisi di tutte le combinazioni è ancora un problema di tipo NP, dunque va risolto per enumerazione dei casi possibili. La prima questione da risolvere è, dato un certo blocco omogeneo, determinare l'insieme di tutte le sue prese possibili. A tal fine si devono determinare, per ogni singolo pezzo, tutti i modi di piazzarlo, da solo, o con altri pezzi a lui adiacenti. Definiamo quindi come *grandezza* di una presa il numero di pezzi che vengono piazzati con la sua applicazione. La grandezza massima di una presa, $MaxPick_{ee}$, sarà stabilita in base alle proprietà dell'end effector e risulterà uguale al numero massimo di pezzi afferrabili con un unico movimento. La presa più piccola potrà avere grandezza unitaria e corrisponderà al collocamento di un unico prodotto dello strato. Per ogni blocco dovranno essere determinate tutte le prese possibili, da quelle di grandezza unitaria a quelle di grandezza $MaxPick_{ee}$. La funzione illustrata in Figura 3.3 svolge il compito di determinare tutte le prese orizzontali, ossia tutte quelle prese i cui prodotti risultino allineati lungo l'asse X. La valutazione delle prese a grandezza unitaria è sempre trattato come caso degenere in quanto la sua risoluzione è banale: basta associare una presa ad ogni singolo prodotto. Per quanto riguarda le prese a grandezza maggiore invece, l'idea è quella di scorrere i vari pezzi del blocco e verificare se

ci sono un numero di pezzi pari alla grandezza della presa, tutti adiacenti tra loro. Per determinare le prese verticali basterà implementare una funzione analoga che applica una ricerca dei pezzi adiacenti lungo l'asse Y.

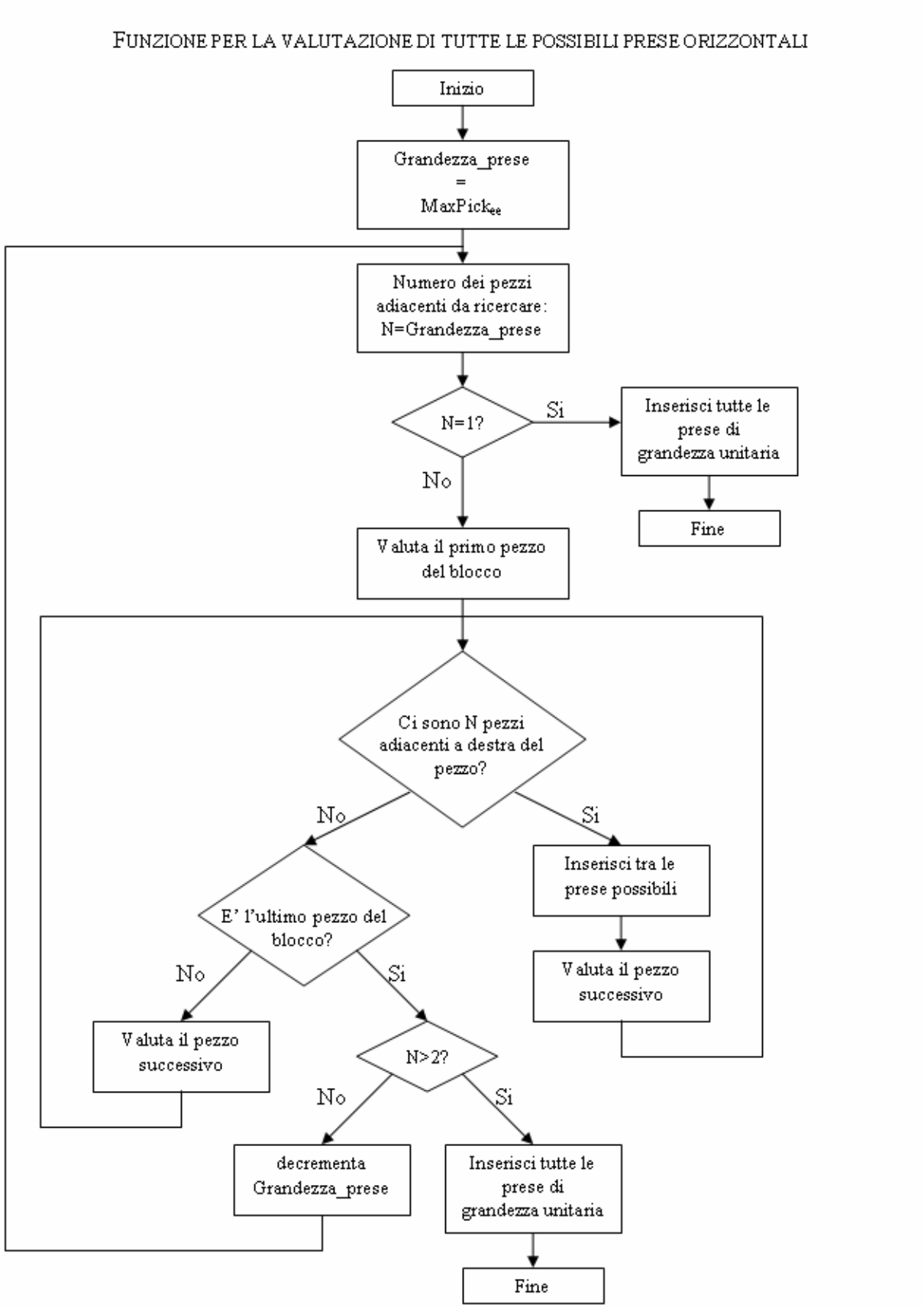


Figura 3.3: ***Funzione per la valutazione di tutte le prese possibili di un sottoblocco***

K.A.Dowsland [20], fornisce una casistica del tipo dei sottoblocchi possibili che potrebbero risultare dai vari layouts: testare l'algoritmo su tutte le casistiche proposte è un buon metodo per determinarne l'efficienza.

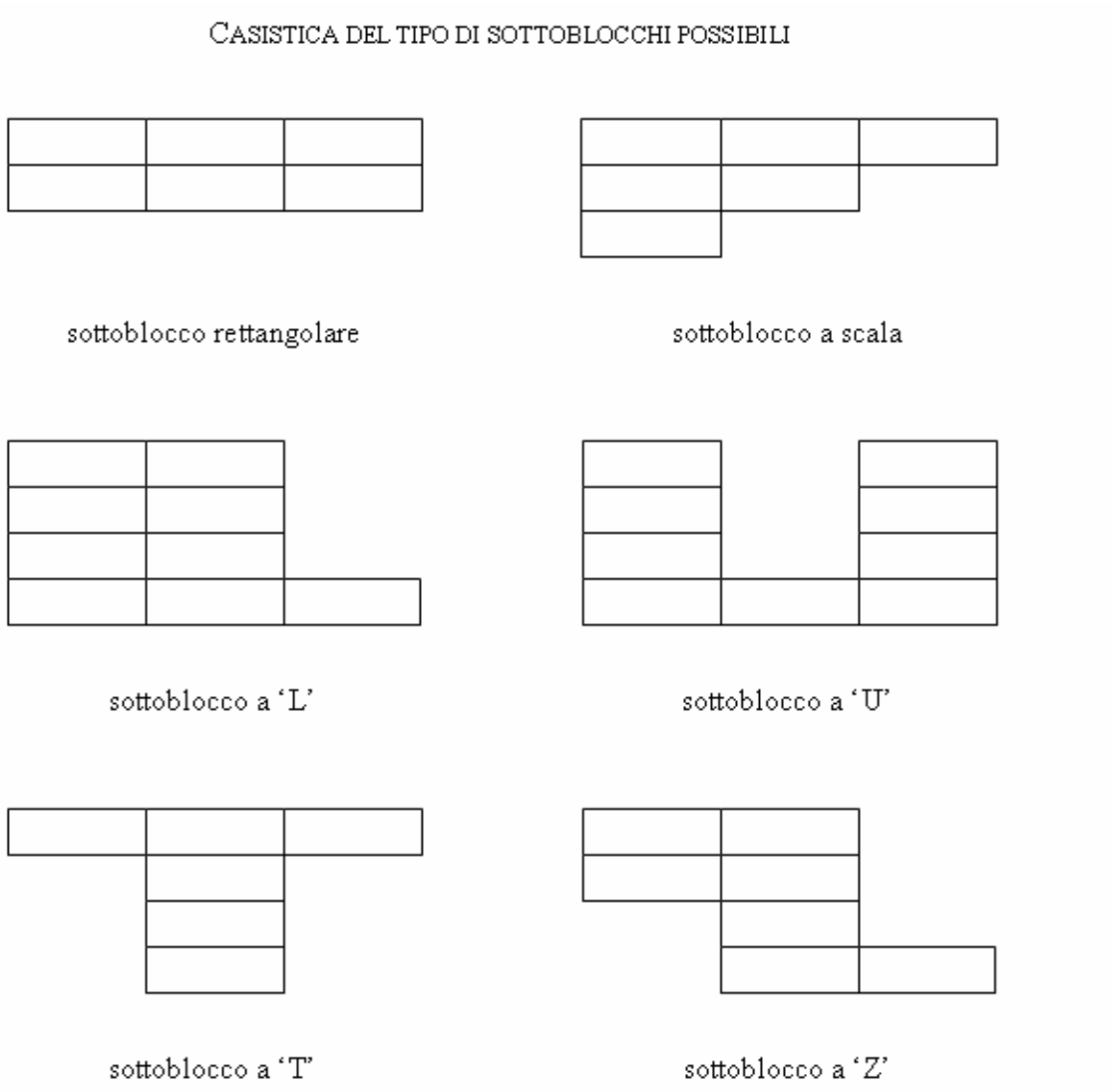


Figura 3.4: possibili sottoblocchi derivanti da un generico layout

Prendiamo ad esempio il sottoblocco a 'Z':

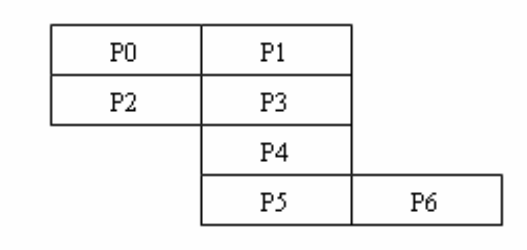


Figura 3.5: sottoblocco di tipo 'Z'

stabilendo la nomenclatura dei pezzi come in Figura 3.5 e che l'end effector possa prendere fino a quattro prodotti con un'unica presa, ad esso sarà associato il seguente insieme di prese possibili:

Lista Prese Possibili : $MaxPick_{ee}= 4$

Prese di grandezza 4 : [P1-P3-P4-P5]

Prese di grandezza 3 : [P1-P2-P3],[P3-P4-P5]

Prese di grandezza 2 : [P0-P1],[P0-P2],[P1-P3],[P2-P3],[P3-P4],[P4-P5],[P5-P6]

Prese di grandezza 1 : [P0], [P1], [P2], [P3], [P4], [P5], [P6]

A questo punto il problema è quello di determinare l'insieme delle prese che andranno effettuate per ricostruire il sottoblocco. Come si può osservare, lo stesso prodotto, nell'esempio è stato evidenziato P1, in generale risulterà appartenente a più prese diverse e ciò dovrà essere tenuto in considerazione quando si vanno a valutare le loro combinazioni: di volta in volta dovranno essere escluse tutte quelle prese contenenti i prodotti già piazzati. Si osservi inoltre che onde non ripetere prese contenenti gli stessi prodotti, la funzione che genera l'insieme delle prese possibili rappresentata dall'algoritmo di Figura 3.3, analizza le adiacenze sempre lungo una certa direzione e mai in quella opposta. In questo modo la presa a grandezza 4 viene trovata soltanto durante la valutazione del pezzo P1 in quanto vede sotto di se, lungo l'asse Y, altri tre pezzi tutti adiacenti tra loro. Al contrario durante l'analisi di P3, la funzione non identifica alcuna presa di ordine 4 dato che sotto di se il pezzo non ha un numero sufficiente di prodotti.

Una volta implementata questa funzione si potrebbe già pensare di realizzare l'algoritmo completo che valuta tutte le possibili combinazioni per determinare la soluzione ottimale a prese minime. Tuttavia senza l'applicazione di opportune regole di taglio che permettano di limitare i calcoli

computazionali, l'algoritmo che ne risulterebbe sarebbe troppo lento. Per migliorarne l'efficienza si possono dedurre opportune regole di taglio dalle seguenti osservazioni:

i. Se un intero blocco può essere collocato con un'unica presa, è inutile andare ad analizzare tutte le possibili varianti: la soluzione a prese minime sarà quella che prende tutti insieme i prodotti del blocco.

ii. Sviluppando una strategia di ricerca di tipo DFS, ad ogni analisi di una nuova presa potranno essere fatte le seguenti verifiche:

   - Stima del lower-bound: se questo è superiore alla soluzione a prese minime già ottenuta, sarà inutile andare a valutare il nodo in questione.
   - Onde evitare di andare a considerare permutazioni di una stessa combinazione di prese, qualora nell'analisi delle varie combinazioni ricapiti una presa già completamente esplorata, risulterà inutile continuare in quella analisi in quanto sarà sicuramente già stata valutata nella precedente esplorazione.

Applicando queste regole di taglio è chiaro che l'algoritmo risulterà tanto più veloce quanto prima si trova una soluzione a prese minime e tanto più accurata risulti la stima del lower bound. Al fine di trovare più velocemente la soluzione a prese minime, sarà utile passare all'algoritmo le prese possibili ordinate per grandezze decrescenti. In questo modo saranno analizzate per prime le prese che inseriscono il numero maggiore di pezzi. Per quanto riguarda il lower bound è chiaro che se N è il numero dei pezzi componenti il blocco, come minimo ci vorranno sempre un numero maggiore o al più uguale a $\left\lceil \frac{N}{MaxPick_{ee}} \right\rceil$ prese; tuttavia si può cercare di migliorare questa approssimazione considerare il numero di righe e di colonne su cui è collocato il blocco: se è disposto su 'r' righe e 'c' colonne, sicuramente sarà impossibile ricomporlo con meno del minimo tra (r,c) prese.

Partendo da queste considerazioni, si può implementare un algoritmo a carattere ricorsivo, come illustrato nella seguente Figura 3.6.

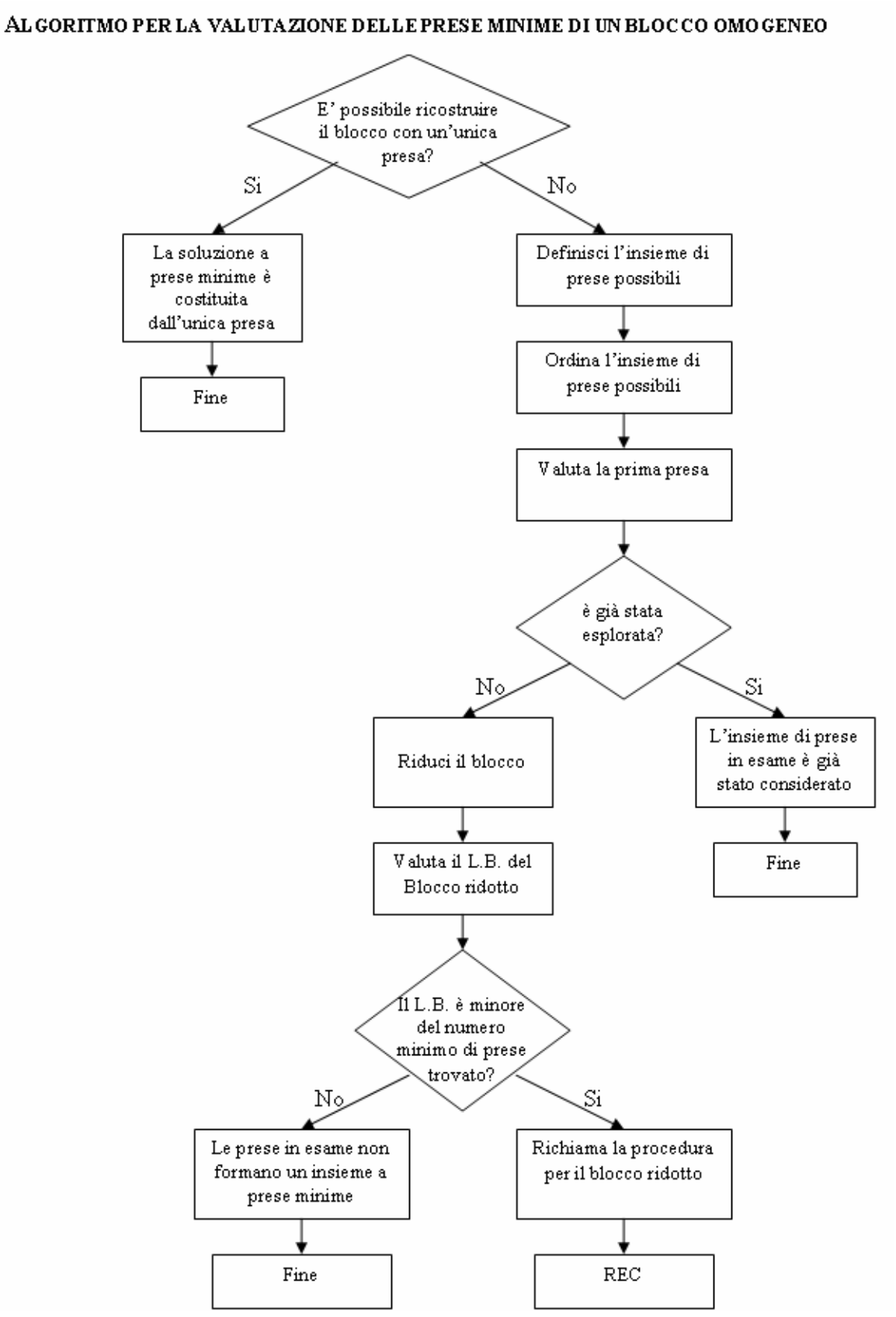


Figura 3.6: ***Algoritmo per la valutazione delle prese minime per un blocco omogeneo***

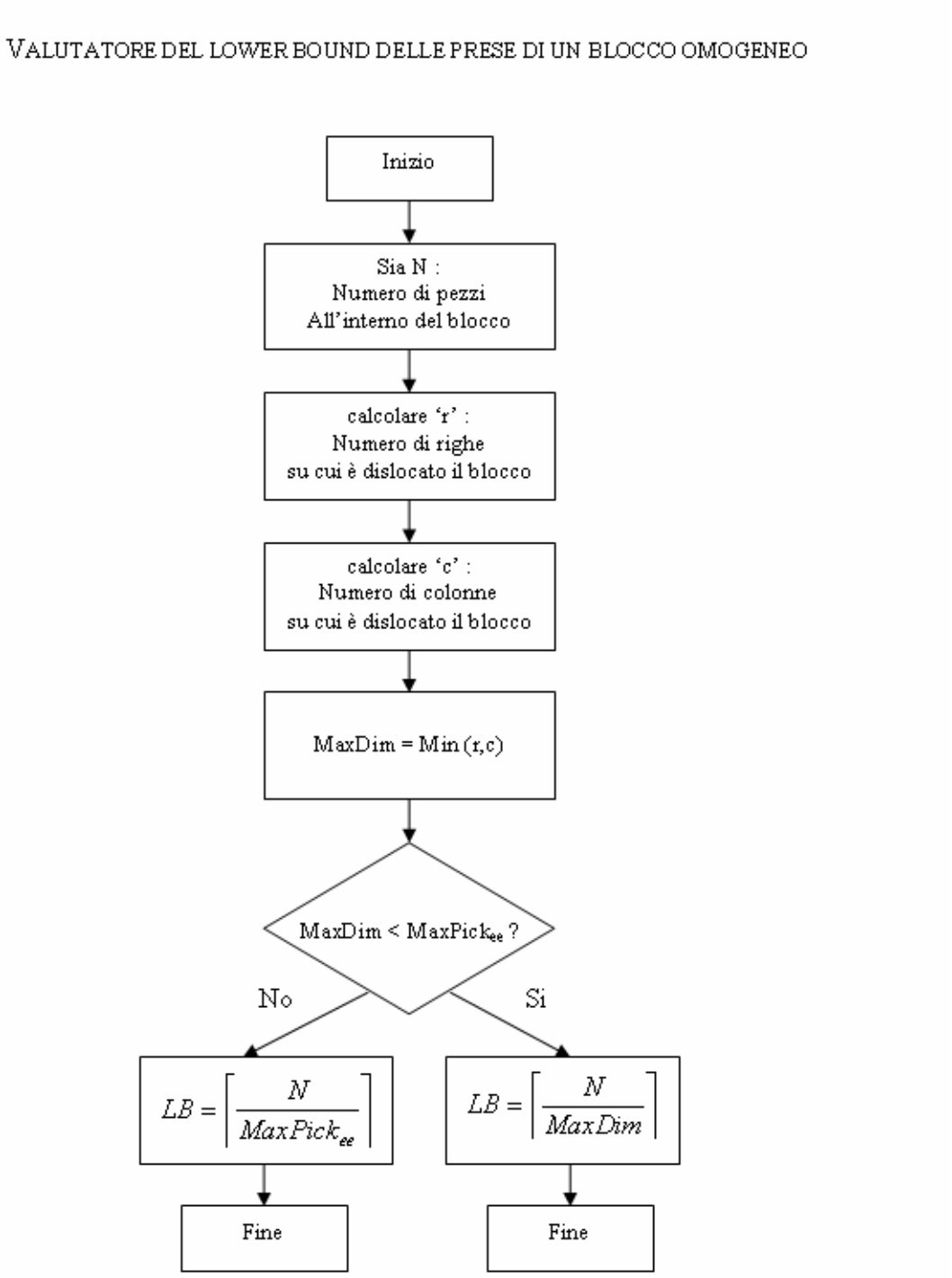


Figura 3.7: ***Algoritmo per la valutazione del lower-bound delle prese***

Mettendo insieme tutte le varie prese minime di ogni singolo sottoblocco omogeneo di un layout si otterrà l'insieme delle prese minime dell'intero strato.

3.2. Utilizzo della linea di trasmissione

Dato un certo layout in versione pezzi, mediante l'algoritmo proposto è possibile determinare i modi di ricostruirlo impiegando il minor numero di prese e rilasci. Sempre riprendendo l'esempio del sottoblocco a 'Z' di Figura 3.5 con $MaxPick_{ee}$= 4, si osserva che il numero minimo di prese impiegabili per la ricostruzione è tre e ci sono due modi di ricostruirlo con tre prese-rilasci:

1° modo : [P1-P3-P4-P5], [P0-P2], [P6]

2° modo : [P1-P3-P4], [P0-P2], [P5-P6]

Ricostruire lo strato con una serie di prese piuttosto che con un'altra non è affatto indifferente in quanto come già spiegato, si devono tenere in considerazione anche i tempi di preparazione dei pezzi al fine linea. Un utilizzo uniforme della linea di trasmissione rappresenta sempre il caso migliore. Nell'esempio citato, il 2° modo dovrebbe essere preferito al primo, in quanto vengono collocati costantemente due o tre prodotti per volta, ossia le prese risultano piuttosto omogenee: in questo modo si può pensare di attuare un accumulo al fine linea pressoché costante. Di conseguenza una volta definiti i vari modi per ricostruire lo strato, tutti a prese minime, possiamo pensare di stimare l'utilizzo della linea mediante un coefficiente che tenga conto della deviazione standard rispetto al numero medio di prodotti per presa teoricamente necessario per ricostruire il layout. Definendo quindi

Countpezzi = numero di prodotti da inserire nello strato

Countprese = numero di prese minime col quale è possibile ricostruire lo strato

$$\overline{P} = \frac{Countpezzi}{Countprese} \qquad \text{teorica presa media} \qquad (1)$$

P_i la grandezza della presa 'i' con i = 1..*Countprese*

la deviazione standard rispetto al numero medio di prese $\overline{P}$ risulterà

$$\sigma = \sqrt{\frac{\sum_{i=1}^{Countprese} (P_i - \overline{P})^2}{Countprese}} \qquad (2)$$

In questo modo possiamo stimare un coefficiente di utilizzo della linea di trasmissione:

$$Coeff_{utilizzolinea} = \frac{1}{1+\sigma} \qquad (3)$$

Più il coefficiente si avvicina a 1 più l'utilizzo della linea di trasmissione risulterà uniforme. Stimando tale coefficiente potremo valutare quale set di prese minime sia più adatto alla ricostruzione dello strato in questione.

3.3. Ordinamento delle Prese

Come detto precedentemente, per realizzare le prese, il robot ha bisogno di avere sempre almeno due lati liberi per poter manovrare. Di conseguenza il set a prese minime scelto va inserito in una lista ordinata di prese tali che, attuandole in sequenza, ogni presa si trovi sempre con almeno due lati liberi per il suo collocamento. Questa è una condizione necessaria per il riordino. La prima cosa da fare è implementare una funzione che verifica quanti lati liberi ci sono per una data presa di un certo set. Al solito però, dato lo stesso set di prese, ci potranno essere diverse sequenze in grado di ricostruire lo strato senza intoppi. In tal caso dovrà essere privilegiata quella che ottimizza meglio la linea di trasmissione. Per un suo utilizzo uniforme, sarebbe bene evitare quelle sequenze che inseriscano prima tutte le prese a grandezza maggiore e poi tutte quelle a grandezza minore: la miglior cosa consisterebbe nel riuscire ad alternare le prese a varia grandezza. In questo modo si avrebbe un utilizzo uniforme della linea di trasmissione non considerando la singola presa, bensì considerando la loro successione temporale. Per realizzare ciò si può pensare di riordinare il set in base alla grandezza delle prese. A questo punto si applicherà una ricerca delle prese con almeno due

lati liberi, scorrendo il set in maniera alternata: scorrendo il set in un verso saranno trovate per prime le prese a grandezza maggiore, scorrendolo nell'altro saranno determinate per prime quelle a grandezza minore. L'algoritmo è illustrato nella Figura 3.8. Si osservi che una volta definita la lista ordinata dei movimenti da eseguire, volendo depallettizzare utilizzando lo stesso robot, potremmo semplicemente ripercorrere tale lista in senso inverso.

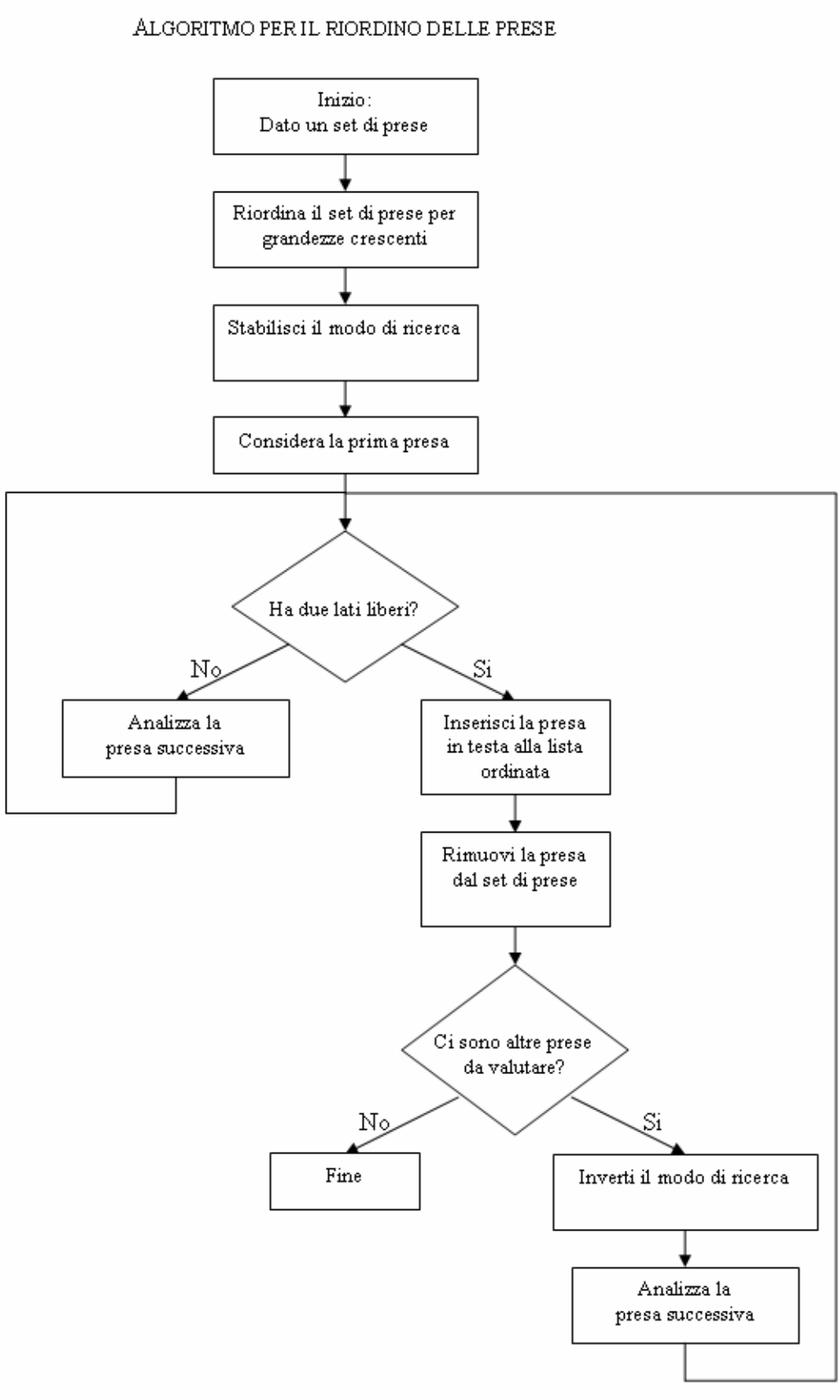


Figura 3.8: ***Algoritmo per il riordino delle prese***

3.3.1. Definizione dell'accosto

Al robot verranno fornite le coordinate spaziali relative all'inserimento dei prodotti di una presa. Tali prodotti vengono trasportati con una certa velocità nei pressi del punto di rilascio e accostati in maniera più lenta. Una volta quantificato l'offset d'accosto necessario al robot per un corretto rilascio, il programma che realizza le prese deve anche determinarne la direzione d'accosto. Se $(\overline{X},\overline{Y},\overline{Z})$ è il punto di inserimento e $(\Delta X,\Delta Y,\Delta Z)$ rappresenta l'offset d'accosto, la direzione d'accosto sarà gestita in base ai segni degli offset che dovranno quindi essere opportunamente stabiliti. A riguardo si devono considerare le seguenti regole:

i. Presenza di tutti e quattro i lati liberi => accosto lungo l'asse 'Z': ΔX=0 , ΔY=0

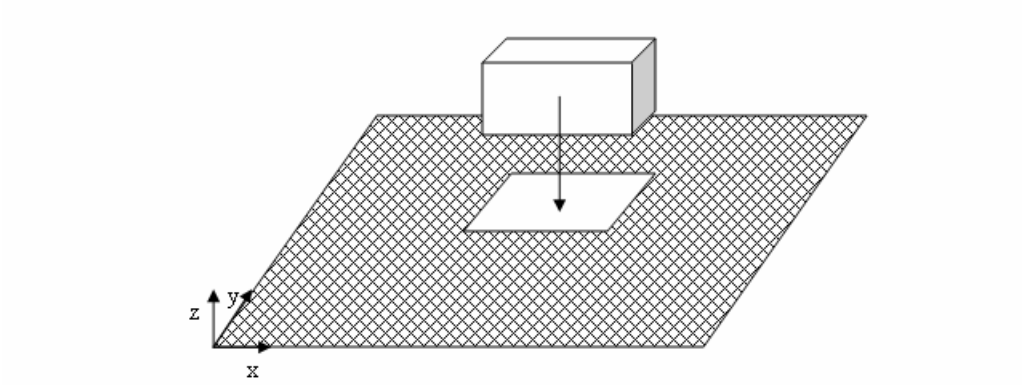


Figura 3.9: accosto verticale

Solitamente questo tipo di accosto si verifica con l'inserimento del primo blocco quando non sono presenti altri prodotti sullo strato. In questo caso l'offset d'accosto sarà banalmente $(0,0,\Delta Z)$

ii. Presenza di tre lati liberi => uno degli offset d'accosto sul piano è nullo: Δx=0 o Δy=0. occorre valutare il segno dell'altro

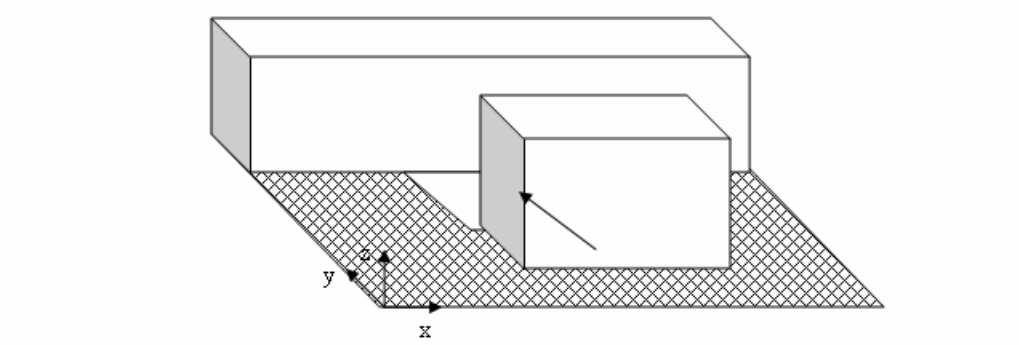


Figura 3.10: accosto con tre lati liberi

Dato il sistema di riferimento di Figura 3.10, l'accosto in questo caso risulta $(0,-\Delta Y,\Delta Z)$

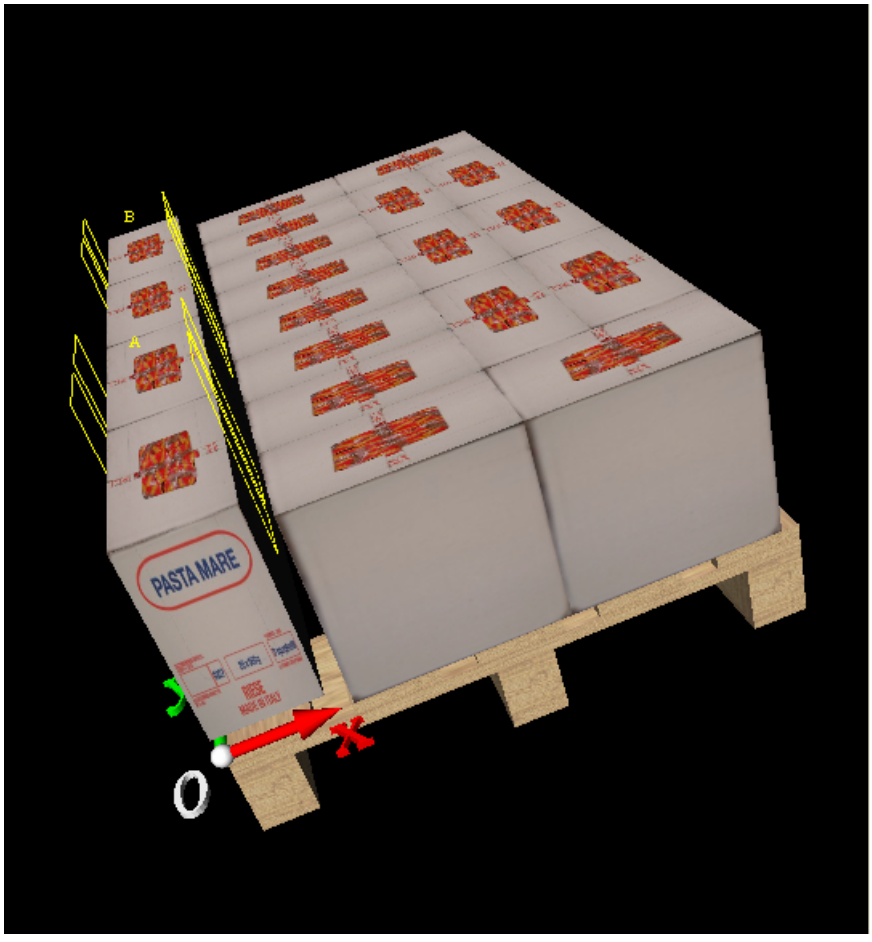


Figura 3.11: visualizzazione di un accosto laterale dal *PALLETWIZ* fornito da *Scienza Machinale*

Secondo il sistema di riferimento di Figura 3.10, l'accosto risulta $(-\Delta X,0,\Delta Z)$

iii. Presenza di due soli lati liberi adiacenti => nessun offset d'accosto è nullo

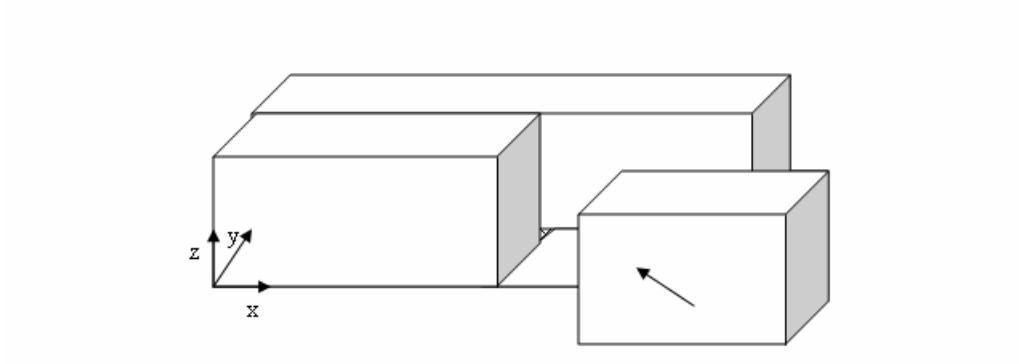


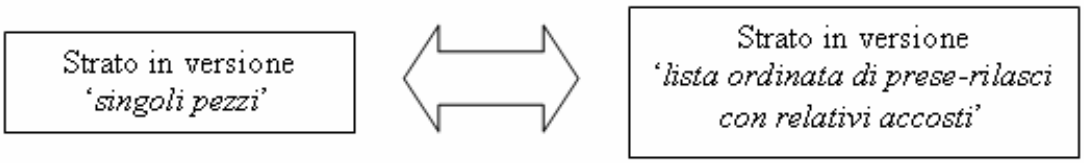
Figura 3.12: accosto con due lati liberi

Questo è il caso più comune: nell'esempio l'accosto è $(+\Delta X, -\Delta Y, \Delta Z)$

Una volta stabilito l'ordinamento delle prese, basterà implementare una funzione che ad ogni inserimento preso in sequenza, valuti quanti e quali sono i lati liberi e assegni di conseguenza il corretto segno e valore dell'offset d'accosto per il rilascio.

3.4. Conclusioni

In conclusione, stimando il coefficiente di utilizzo della linea di trasmissione, siamo in grado di ristabilire una corrispondenza biunivoca tra layout in visione pezzi e layout in visione prese in modo che ad ogni layout prodotto da GeneraLayers sia associato un unico set di prese minime. Tenendo in considerazione le esigenze del robot pallettizzatore e le caratteristiche della linea di trasmissione, tale set di prese può essere opportunamente riordinato per produrre la sequenza temporale di ricostruzione. In questo modo è possibile, dato un qualunque layout in visione pezzi, effettuare una vera e propria conversione nella sua versione ordinata di prese-rilasci con i relativi accosti.



Questa proprietà è molto importante in quanto stabilisce una vera e propria corrispondenza biunivoca tra le diverse visioni dei layers. Adottando questa strategia diviene possibile associare ad ogni layers in visione pezzi uno ed un solo layer in visione prese.

Capitolo 4 : Stabilità

Una volta stabiliti i layouts che inseriscono il numero massimo di pezzi per strato, resta da valutare come sovrapporli l'uno con l'altro in modo da formare una pila stabile. La stabilità va intesa sia in senso statico che dinamico, in quanto il pallet deve essere stare in piedi durante la sua costruzione ma deve anche sopportare eventuali forze dinamiche durante il suo trasporto. Essa solitamente è un parametro piuttosto variabile, nel senso che non dipende esclusivamente dal sistema di pallettizzazione, ma può variare fortemente anche in funzione del tipo di prodotto da pallettizzare ed eventualmente dalle particolari esigenze dell'utente. Di conseguenza per la valutazione della stabilità vengono considerati dei criteri generali con dei parametri da aggiustare a seconda dei casi.
Secondo H.Carpenter e W.B.Dowsland [23] i criteri di stabilità da tenere in considerazione sono i seguenti:

1. Ciascun prodotto dello strato superiore deve avere la sua superficie di base a contatto con almeno altri due prodotto dello strato sottostante. Tale contatto deve essere stimato con una certa percentuale %X, sotto la quale il contatto deve essere considerato inefficace.
2. Ciascun prodotto dello strato superiore deve avere una certa percentuale %Y della propria superficie di base a contatto con lo strato inferiore.
3. Non devono esserci tagli dritti (guillotine cuts) di una certa dimensione I, che attraversino più di una certa percentuale %Z della massima lunghezza della pila.

Il criterio 1 serve per evitare la formazione di colonne di pezzi che risulterebbero altamente instabili sia durante l'inserimento dei prodotti stessi, sia durante il trasporto del pallet. Il criterio 2 evita eventuali collassi dei singoli prodotti già al momento della collocazione. Mentre il criterio 3 serve ad evitare eventuali tagli a ghigliottina o sgradevoli buchi di particolari dimensioni. I tagli a ghigliottina nella valutazione della stabilità possono essere dannosi per varie ragioni: innanzi tutto, se non c'è una sufficiente copertura, potrebbero verificarsi delle pericolose colonne di blocchi che

rischierebbero di produrre dei problemi di stabilità simili a quelli inerenti alle più semplici colonne di pezzi in quanto potrebbero far aprire e collassare il pallet; in secondo luogo durante i trasporti i pezzi potrebbero muoversi scivolando lungo il taglio facendo perdere la stabilità del sistema. Analoghe considerazioni possono essere fatte per la presenza di buchi di dimensioni particolarmente grandi all'interno del layout in quanto, durante il trasporto, alcuni pezzi soggetti alle forze dinamiche, potrebbero traslare al loro interno con eventuale collassamento del sistema. Secondo queste definizioni, due strati tali che posto uno sull'altro, soddisfino questi tre criteri, risultano stabili. Si può cercare di dare una stima generale per le percentuali da adottare: ad esempio nel citato articolo si suggerisce di stimare tra il 5% e il 15% la superficie di contatto minima per il criterio 1, non meno dell' l'80% per la superficie d'appoggio del criterio 2 e si sconsiglia di utilizzare layout con tagli a ghigliottina che lo attraversino per il 100% della sua lunghezza. Naturalmente un utente potrebbe pensare di modificare le varie percentuali X,Y,Z a seconda delle proprie esigenze. Tuttavia in molti casi i suddetti criteri così come sono stati impostati, risulterebbero alquanto limitativi. A tal fine Leipala e Nevalainen [26] hanno cercato di rivederli apportando le seguenti modifiche:

Variazione criterio 1 : La formazione di colonne durante l'impilamento è particolarmente pericolosa qualora si formi troppo a ridosso dell'esterno del degli estremi dell'area pallettizzata. Il suggerimento è quindi di considerare il criterio 1 non su tutta la superficie pallettizzata bensì lungo la fascia perimetrale esterna di una certa ampiezza W.

Variazione criterio 3 : I tagli a ghigliottina sono una proprietà del singolo strato che non tengono conto della possibile ricoesione che potrebbe fornire lo strato superiore. Di conseguenza in presenza di tagli a ghigliottina, può essere utile considerare la copertura offerta dallo strato superiore. Se i pezzi dello strato superiore ricoprono una certa %Z' del taglio, allora il sistema non deve essere considerato instabile.

Tali variazioni, grazie all'inserimento di ulteriori parametri, permettono un affinamento dei criteri prima esposti rendendoli in questo modo meno stringenti e magari più adatti per un maggior numero di applicazioni pratiche.

Il problema di questo approccio alla stabilità è che i criteri proposti vengono analizzati tutti contemporaneamente, senza stabilire una priorità tra di essi. Consideriamo ad esempio il criterio 2: la sua verifica appare di fondamentale importanza per la stabilità finale del sistema. Senza di essa infatti il pallet potrebbe non avere la capacità di autosostenersi, in quanto verrebbe a mancare il necessario appoggio fisico per gli strati superiori su quelli inferiori; mentre in mancanza degli altri criteri, si potrebbe pensare di recuperare la stabilità mediante interventi esterni come l'inserimento di interfalde o l'utilizzo di particolari colle adesive. Da questo punto di vista potrebbe essere interessante affrontare il problema della stabilità suddividendolo in due fasi distinte: una prima fase che controlli la capacità di autosostenersi del pallet e una seconda fase che controlli se sono necessari degli ulteriori accorgimenti esterni per recuperare la stabilità finale del sistema. In questo modo i criteri analizzati nella prima fase risulteranno condizioni necessarie alla stabilità del sistema, mentre quelli analizzati nella seconda fase risulteranno condizioni opzionali la cui verifica potrà stabilire quali ulteriori accorgimenti esterni siano necessari. Ovviamente il passaggio alla seconda fase avrà senso solo se i criteri della prima fase siano stati verificati con esito positivo.

4.1. Condizioni necessarie

E' chiaro che per avere stabilità di uno strato sopra un altro occorre che tutti i prodotti dello strato superiore abbiano una posizione stabile. Tuttavia non ci possiamo accontentare che il baricentro del prodotto appoggi sulla superficie sottostante perché sarebbe una condizione notevolmente precaria dal punto di vista reale. La condizione imposta dal secondo criterio di Carpenter e W.B.Dowsland è da questo punto di vista, una buona condizione per la stabilità. In effetti prendendo una copertura d'appoggio dell'ordine dell'80-90% (Leipala e Nevalainen suggeriscono addirittura una copertura del 95%) della superficie di base del prodotto, sarà improbabile che la collocazione del prodotto non

risulti stabile. Questa condizione risulta appropriata anche nel caso che i prodotti da pallettizzare non siano dei corpi prettamente rigidi, ma potrebbero anche essere ad esempio dei corpi molli, tipo dei sacchi di farina, le cui forme siano comunque approssimabili con dei parallelepipedi. Se invece i prodotti sono delle scatole sufficientemente rigide ed in grado di sostenere il peso degli eventuali strati soprastanti, per confermarne la stabilità potrebbe essere sufficiente controllare che ci sia una certa presa d'appoggio sui quattro angoli della superficie di base del prodotto, vedi Figura 4.1

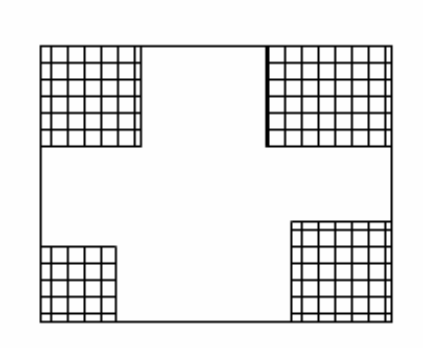


Figura 4.1: stabilità di un corpo rigido

In simili casi, anche se la base del prodotto non appoggia su una grande percentuale della propria superficie totale, un sufficiente appoggio sui quattro angoli garantisce la stabilità. Se il corpo fosse molle finirebbe per collassare al centro e quindi tale criterio non andrebbe bene. In effetti questi programmi sono spesso utilizzati anche per determinare i layouts di prodotti che solo in prima approssimazione hanno dei corpi a forma di parallelepipedo come ad esempio dei sacchi molli. In simili casi la valutazione della stabilità è assai più stringente. Tuttavia potendo contare su un'eventuale interazione con l'utente esterno, nella valutazione della stabilità potremmo distinguere tra prodotti rigidi e prodotti molli: in questo modo la condizione necessaria alla stabilità potrà essere affinata a seconda della situazione.

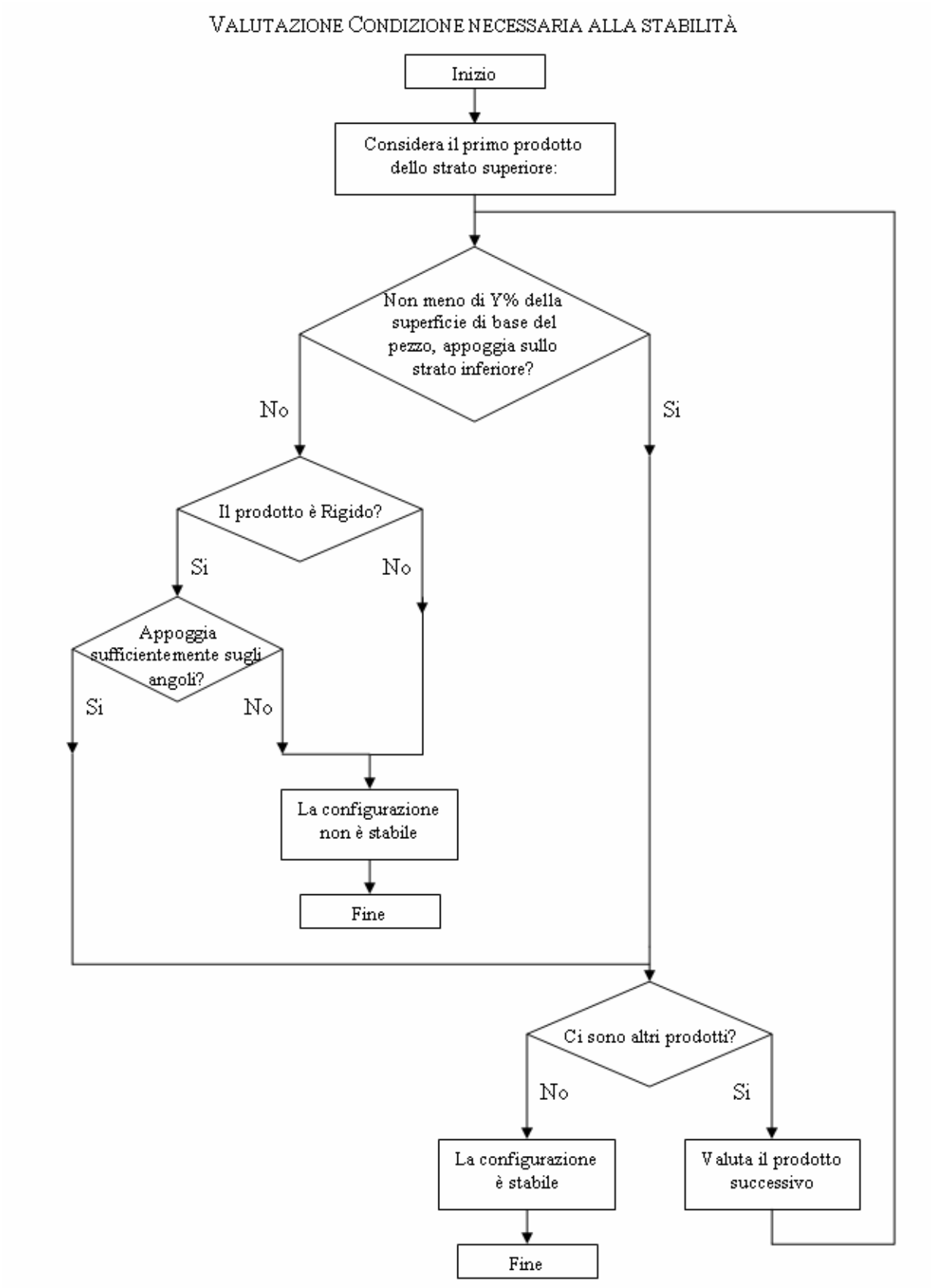


Figura 4.2: ***Algoritmo per le condizioni necessarie alla stabilità***

4.2. Condizioni opzionali

Se si vuole evitare l'utilizzo di interfalde o di colle adesive con evidenti risparmi sui costi e sui tempi di produzione, non ci si può limitare all'analisi delle sole condizioni necessarie, ma si deve procedere negli ulteriori controlli verificando che, almeno entro una certa fascia esterna, non si formino colonne di pezzi, che i tagli a ghigliottina, se presenti, abbiano almeno una copertura sufficiente, e che non ci siano buchi indesiderati. Secondo la definizione del criterio 3 di Carpenter e W.B.Dowsland il taglio a ghigliottina sarebbe soltanto un caso degenere di un buco passante da una parte all'altra del pallet di dimensione nulla (I=0), tuttavia per l'utente potrebbe essere comodo poter distinguere tra i due. Se ad esempio il problema fosse soltanto quello di evitare lo slittamento dei pezzi lungo il taglio, si potrebbe pensare di utilizzare delle colle adesive per tenere insieme i pezzi, cosa di difficile attuazione in presenza di un vero e proprio buco di ampiezza finita, in quanto verrebbe a mancare il contatto diretto tra i pezzi in questione. Quindi nel programma elaborato saranno considerati in maniera distinta i buchi e i tagli a ghigliottina. Per quanto riguarda i parametri per la valutazione delle stabilità saranno dati di default, ma sempre con l'idea che l'utente possa modificarli a seconda delle proprie esigenze. Così ad esempio, se W rappresenta la lunghezza del lato corto della base del prodotto e L rappresenta il lato lungo, il controllo della fascia esterna di default sarà valutata con le dimensioni W+L in modo che verso l'esterno ci sia sicuramente una barriere di pezzi pressoché stabili.

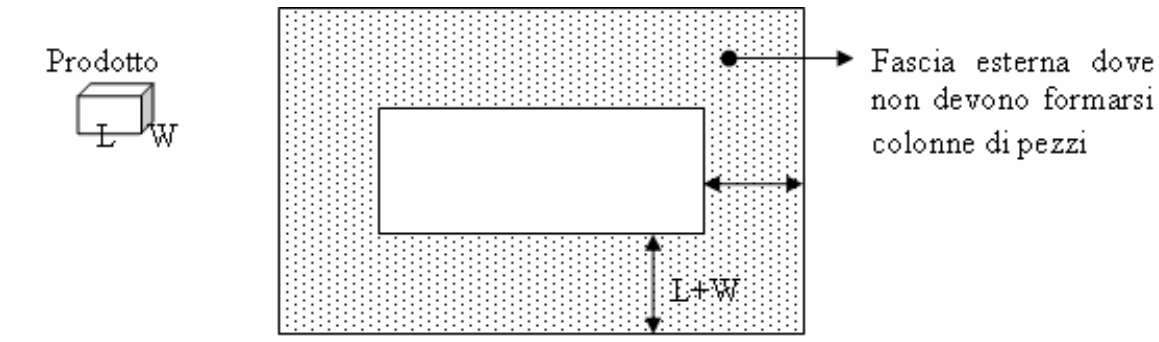


Figura 4.3: fascia esterna stabile

Non avrebbe molto senso valutare un'ampiezza di fascia stabile inferiore a W in quanto si potrebbero formare delle colonne di pezzi fino sull'esterno del pallet. Analoghe considerazioni possono essere fatte per i buchi: considerare un buco di lunghezza inferiore a W non avrebbe valore perché tutto sommato, è naturale che ci siano.

4.3. Implementazione delle funzioni necessarie

In definitiva quindi per l'analisi della stabilità abbiamo bisogno di varie funzioni che eseguano i seguenti controlli:

1° Fase: Controllo dei criteri necessari per la stabilità

- Controllo della superficie d'appoggio di ogni singolo pezzo dello strato superiore
- Controllo dell'appoggio degli angoli della superficie di base del prodotto

2° Fase: Controllo dei criteri opzionali

- Controlla la mancata formazione di colonne di pezzi
- Controlla la presenza di eventuali tagli a ghigliottina
- Controlla l'assenza di eventuali buchi di particolari dimensioni

In questo modo sarà possibile stabilire vari livelli di stabilità: due strati che superino i controlli della prima fase avranno certamente una stabilità sufficiente; qualora superino gli ulteriori controlli della seconda fase gli verranno assegnati dei valori che indichino una migliore stabilità, fino ad arrivare ad una valutazione ottima data dal soddisfacimento dei criteri di Carpenter e W.B.Dowsland che sono i più stringenti.

Passiamo in rassegna i vari algoritmi per la realizzazione delle suddette funzioni.

Per prima cosa ci occupiamo delle condizioni necessarie per le quali si deve calcolare la superficie d'appoggio di ogni singolo prodotto dello strato superiore.

Sia *P1* il pezzo dello strato superiore di cui si vuole valutare l'appoggio e sia *S1* la proiezione della sua superficie di base sul piano XY.

Sia P_i un generico pezzo dello strato inferiore e S_i la proiezione della sua base sempre sul paino XY. La superficie d'appoggio di P_1 su P_i sarà data dall'intersezione delle loro proiezioni:

$$S_{Appoggio_i} = (S_1 \cap S_i) \qquad (1)$$

E d'altronde la superficie d'appoggio totale sarà data dalla somma di tutte le superfici d'appoggio tra S_1 e tutti i vari S_i dello strato inferiore:

$$S_{Appoggio} = \sum_i S_i \qquad (2)$$

L'implementazione di un simile procedimento è illustrato dal diagramma della seguente Figura 4.4

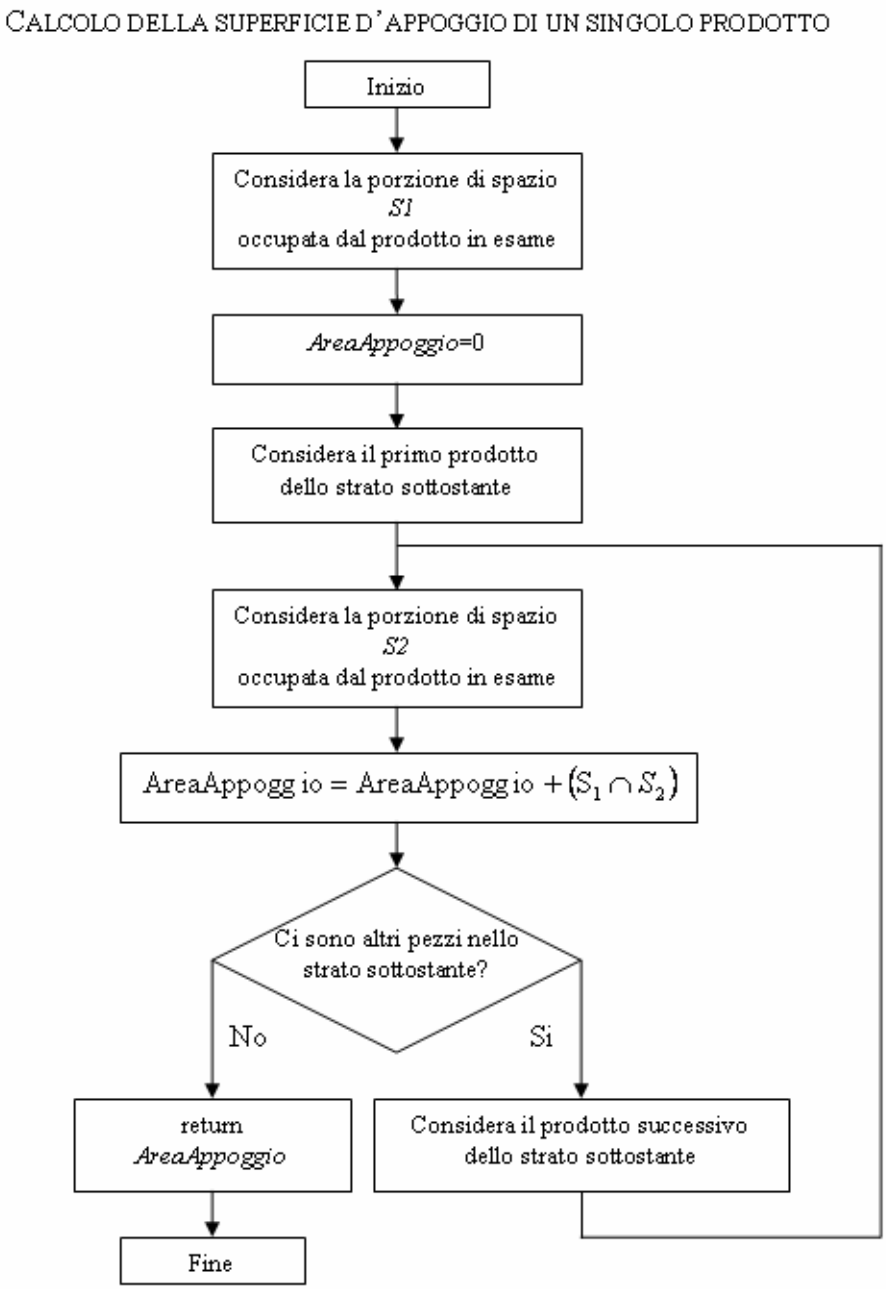


Figura 4.4: ***Funzione che calcola la superficie d'appoggio di un prodotto***

Il cuore dell'algoritmo è quindi costituito da una funzione che, date due porzioni di spazio S1 e S2, stimi la loro eventuale intersezione

Per la verifica della condizione necessaria non occorrerà scorrere sempre tutti i prodotti dello strato sottostante in quanto l'algoritmo potrà essere arrestato non appena *AreaAppoggio* raggiunge il valore richiesto per il soddisfacimento della condizione.

Una volta implementata la funzione che restituisce l'area di intersezione tra due generiche superfici, lo stesso algoritmo di Figura 4.4 viene applicato per la valutazione dell'eventuale appoggio degli angoli del prodotto: basta sostituire a *S1* la porzione di spazio occupata dalla superficie d'angolo e ripetere il controllo su tutti e quattro gli angoli.

La stessa funzione è utilizzata anche per il controllo dell'eventuale presenza di colonne di pezzi il cui algoritmo è rappresentato in Figura 4.5. Nel caso che interessi la stabilità di una certa fascia tale algoritmo non andrà avviato per tutti i prodotti dello strato sovrastante ma soltanto per quelli collocati nella fascia di interesse.

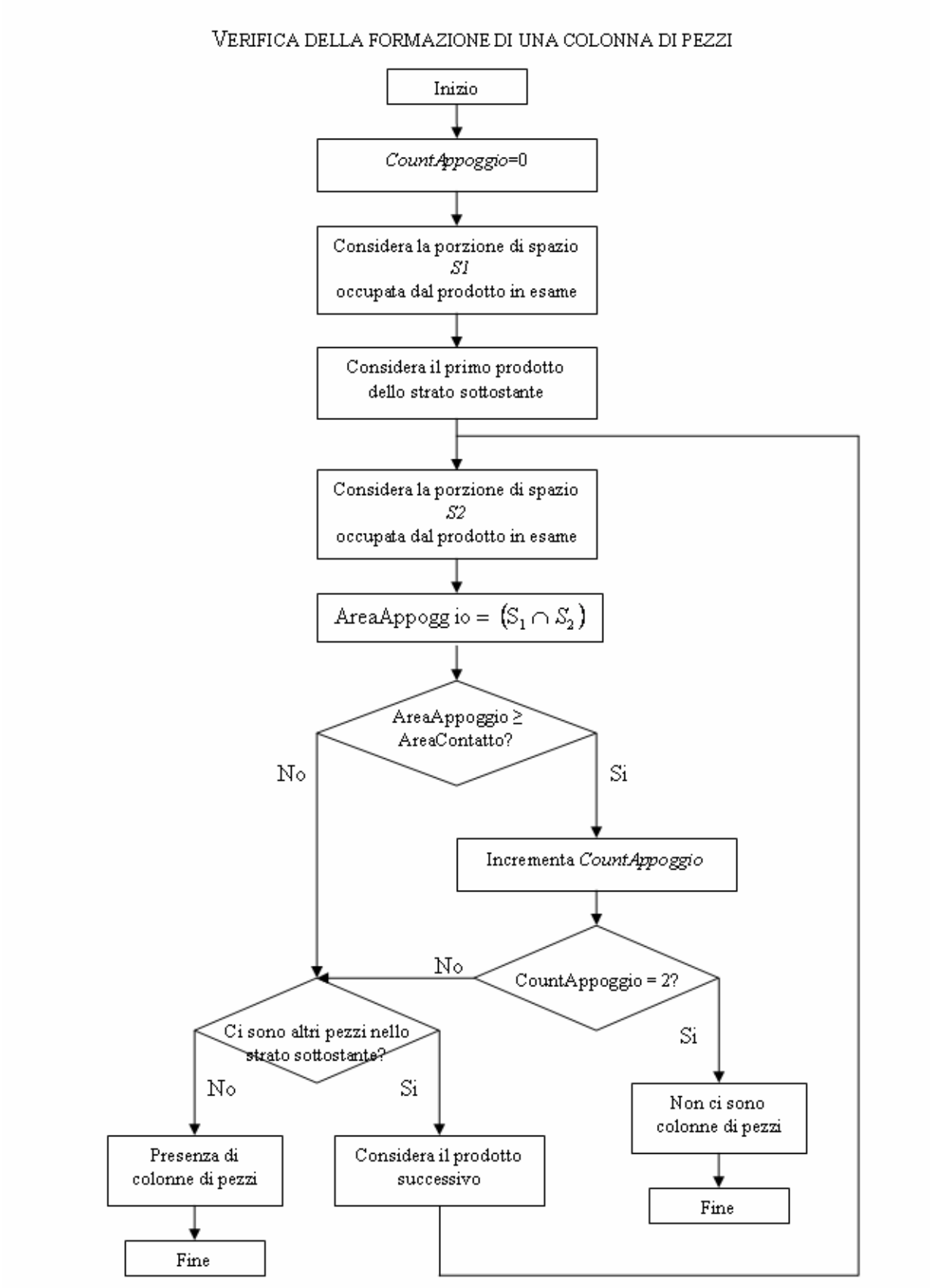


Figura 4.5: ***Funzione che valuta l'eventuale formazione di colonne di pezzi***

La stabilità è un parametro che riguarda la sovrapposizione di due strati. Tuttavia i tagli a ghigliottina e i buchi sono proprietà dei singoli strati che possono essere già valutate in fase di ricostruzione del layout. Per quanto riguarda l'identificazione dei tagli a ghigliottina, l'idea è quella di considerare in maniera distinta i tagli orizzontali e quelli verticali. In riferimento ai tagli orizzontali si avrà un taglio a ghigliottina lungo una certa ordinata Y* se nessuno prodotto dello strato contiene al suo interno l'ordinata Y* in questione. Allo stesso modo si avrà un taglio a ghigliottina verticale lungo una certa ascissa X* se nessun pezzo dello strato la contiene al suo interno. Implementare una funzione che verifichi se un prodotto occupi o meno una certa ascissa o una certa ordinata è banale, tuttavia non avrebbe molto senso controllare la presenza di eventuali tagli lungo tutte le ascisse e tutte le ordinate del pallet. Nel caso di un pancale di dimensioni 1200x1200, volendo assumere una precisione dell'ordine del mm, si dovrebbero infatti impostare 2400 verifiche. Si può cercare di sfruttare il controllo sul pezzo per eliminare le ascisse o le ordinate in cui l'assenza dei tagli a ghigliottina è sicura: chiaramente tali ordinate e ascisse sono quelle occupate dal pezzo stesso. In questo modo ad ogni controllo sul pezzo si potrà sfoltire il numero di tagli da verificare. La Figura 4.6 mostra un algoritmo che adotta questa soluzione. Un altro algoritmo del tutto analogo va implementato per verificare la presenza di eventuali tagli a ghigliottina orizzontali. Una volta analizzato uno strato secondo questi due algoritmi, ad esso saranno associate due liste: una per i tagli orizzontali e una per quelli verticali. All'interno delle liste ogni taglio sarà identificato dalla propria ordinata o ascissa a seconda che esso sia di tipo orizzontale o verticale. A questo punto per valutare la stabilità tra due strati che si sovrappongono, si dovrà valutare la copertura che lo strato superiore offre sui tagli a ghigliottina dello strato inferiore. Ciò può essere valutato considerando quanti pezzi dello strato superiore ricoprono l'ascissa o l'ordinata caratterizzante il taglio dello strato inferiore. L'algoritmo di Figura 4.7 visualizza il procedimento. Alla fine dell'analisi viene fornita la percentuale di ricopertura del taglio. Si osservi che tale algoritmo vale per un singolo taglio dello strato inferiore:ciò significa che l'analisi va ripetuta per tutti i tagli a ghigliottina presenti.

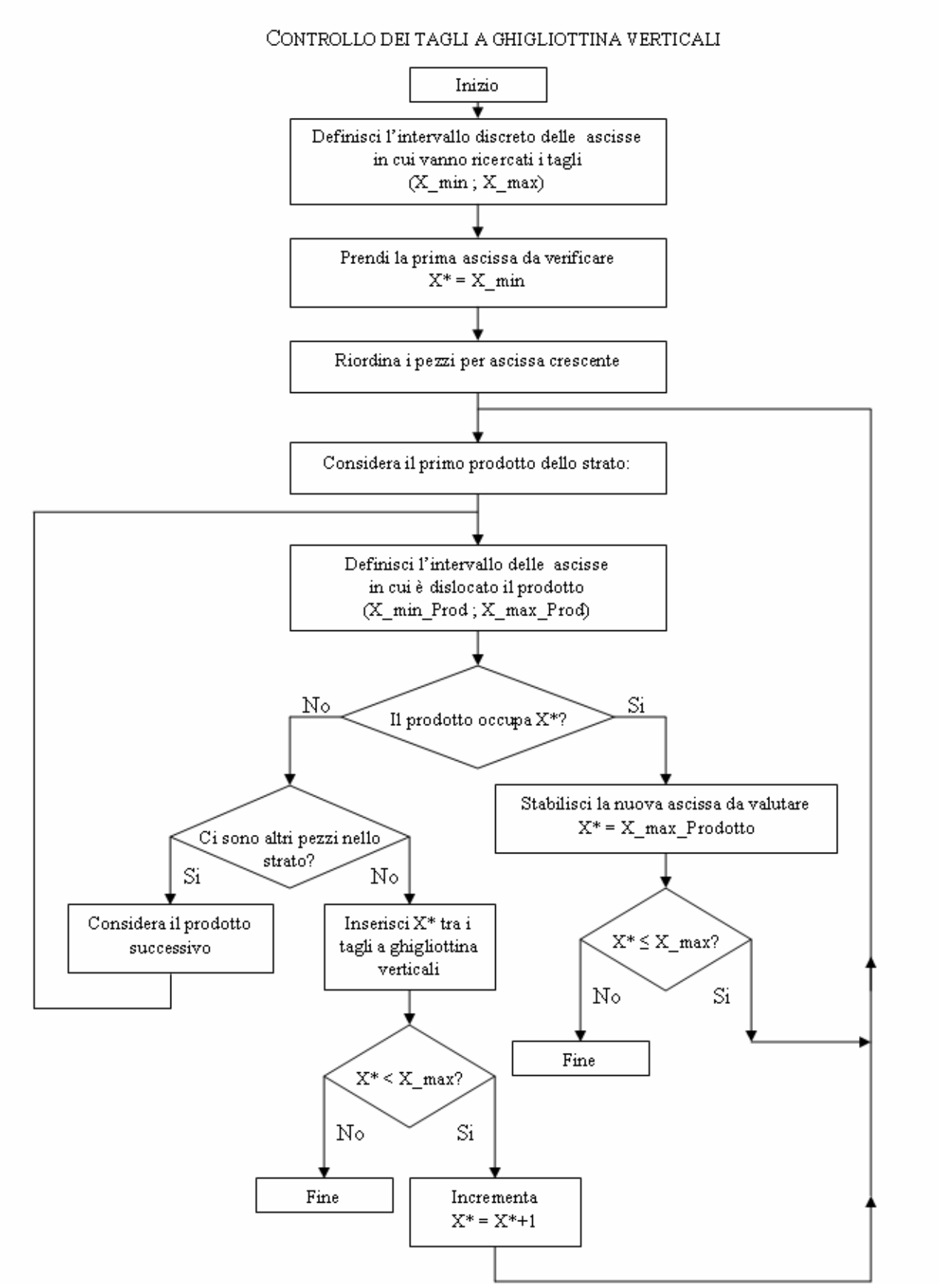


Figura 4.6: ***Algoritmo per la valutazione dei tagli a ghigliottina verticali***

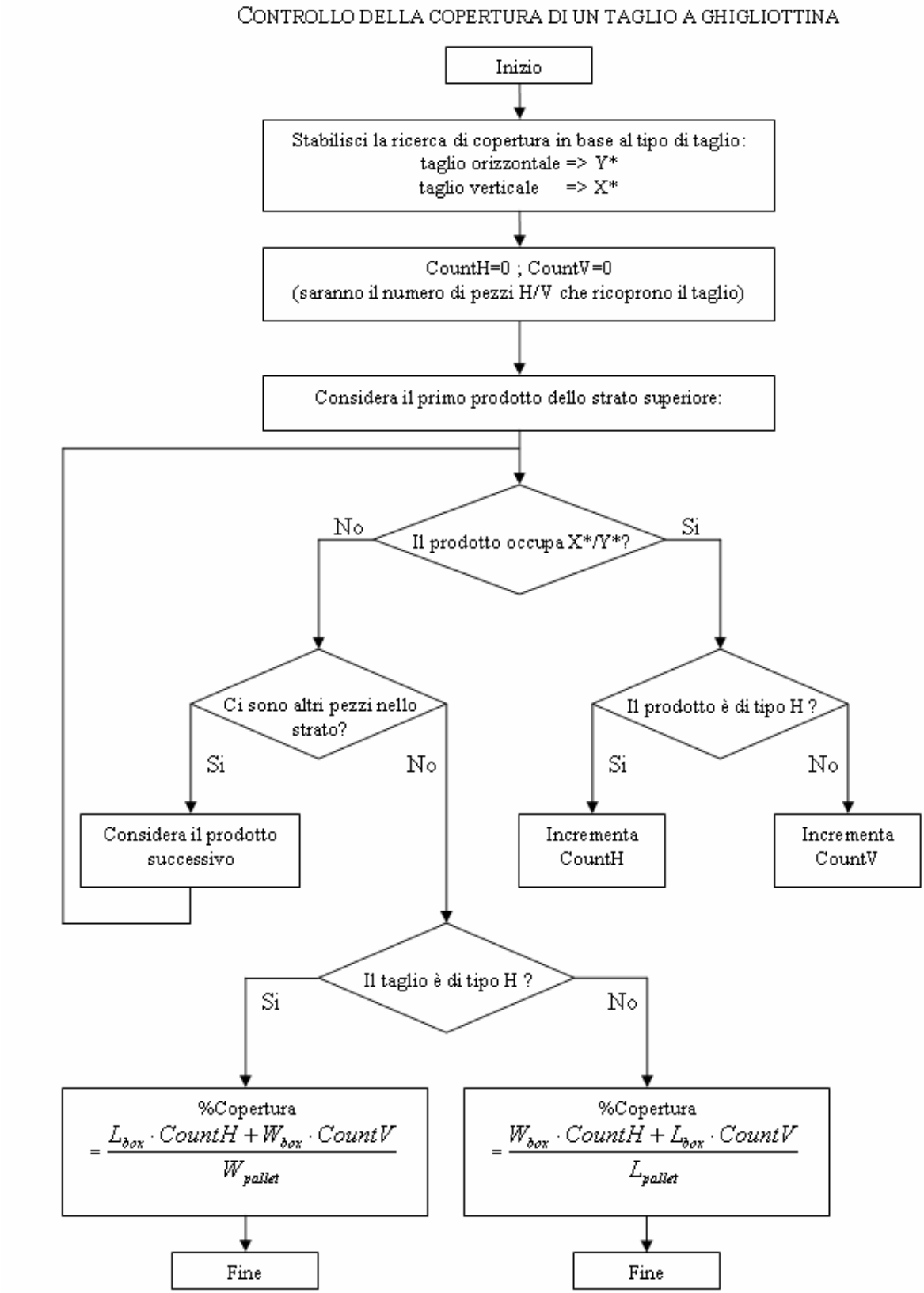


Figura 4.7: ***Algoritmo per la valutazione della copertura di un taglio a ghigliottina***

Per quanto riguarda la presenza di buchi la strada da seguire è un po' diversa. Innanzi tutto non avrebbe senso ricercare tutti i possibili buchi all'interno di un layout, perché in generale ve ne saranno di diversi e di diversa dimensione. Implementare una funzione che riesca a stimare forme e dimensioni dei buchi presenti potrebbe risultare piuttosto dispendioso. Tuttavia per l'analisi della stabilità dovrebbe essere sufficiente una funzione che verifichi la presenza o meno di buchi con dimensioni maggiori o uguali a quelli ritenuti dannosi. Una simile funzione può essere ricavata direttamente da quelle già implementate per l'inserimento dei pezzi durante la ricostruzione del layout: infatti possiamo pensare il buco come un altro pezzo da inserire con determinate dimensioni. In questo modo scorrendo il layout, se riusciamo nell'inserimento, significa che sullo strato c'è spazio sufficiente e quindi è presente un buco di dimensioni maggiori o uguali a quelle indesiderate. Si osservi che scorrere il layout non significa valutare tutte le possibili coordinate del piano del pallet in quanto basterà considerare tutte le varie decision length come definite nello studio di Bhattcharya [15]. In questo modo l'analisi computazionale è notevolmente ridotta. In Figura 4.8 è mostrato l'algoritmo per il riconoscimento di buchi orizzontali. Al solito occorre implementare anche la versione simmetrica per il riconoscimento di quelli verticali.

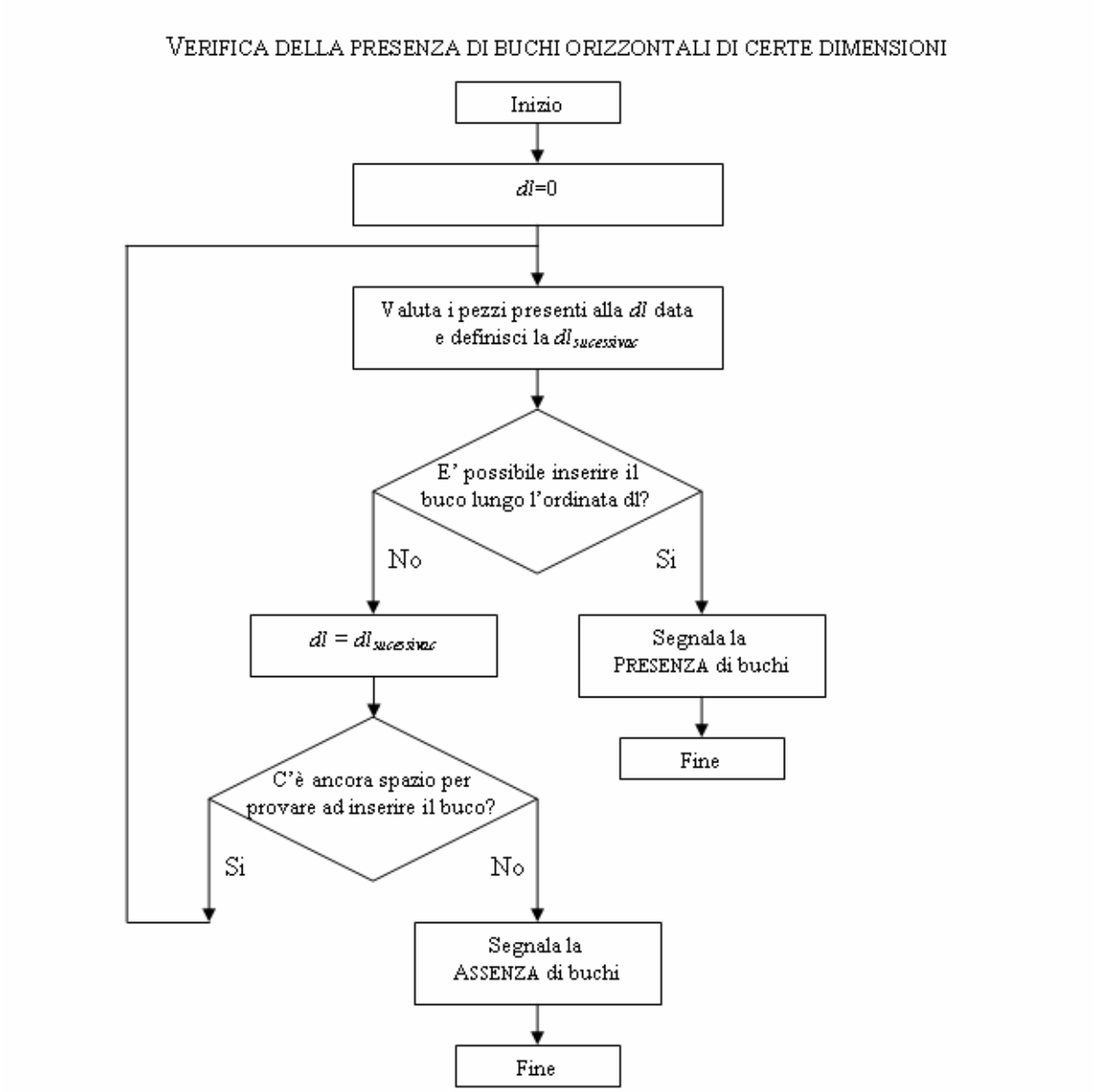


Figura 4.8: ***Algoritmo per la verifica della presenza di eventuali buchi orizzontali***

Una volta implementate tutte queste funzioni si dovrà passare a considerare le combinazioni dei vari strati possibili, valutandone di volta in volta la stabilità, per definire qual è il modo migliore di sovrapporli l'uno sull'altro.

4.4. Strato pari e strato dispari

Per arrivare ad avere una ottima stabilità è chiaro che non sarà possibile realizzare una pallettizzazione ripetendo sempre lo stesso strato, magari quello a prese minime che ottimizza meglio la linea di trasmissione. Sicuramente le condizioni necessarie sarebbero sempre verificate perchè ogni prodotto appoggerebbe sul suo corrispondente dello strato inferiore: in questo modo avrebbe certamente una copertura d'appoggio pari al 100% della sua superficie di base, ma purtroppo si formerebbero colonne di pezzi su ogni prodotto e come minimo sarebbe sempre necessario l'utilizzo dell'interfalda. Avendo a disposizione più layouts diversi che inseriscono comunque lo stesso numero massimo di prodotti, si potrebbe pensare ad alternarli tutti in vario modo sempre cercando di massimizzare la stabilità finale del pallet. Questo procedimento è stato suggerito da Fuh-Hwa Liu and C-J Hsiao [25]. Tuttavia la maggior parte dei sistemi di pallettizzazione opta per alternare due soli strati diversi assegnando un layout per lo strato pari ed uno per lo strato dispari: l'impilamento avviene alternando i due tipi di strato. La ragione di questo approccio fonda la sua bontà sul fatto che: se lo strato pari è stabile sullo strato dispari secondo i criteri prima enunciati, e lo strato dispari è stabile su quello pari, è chiaro che la pila che verrà a formarsi sarà sicuramente stabile. Questo è il metodo che seguiremo.

4.5. Valutazione della stabilità

I criteri esposti servono per dare una valutazione della stabilità di due strati di cui uno è il superiore, e l'altro è l'inferiore. La stabilità di uno strato sull'altro verrà stimata nel seguente modo:

Stabilità = 0	=>	mancata verifica delle condizioni necessarie
Stabilità = 4	=>	verifica delle condizioni necessarie, mancata verifica delle opzionali
Stabilità = 8	=>	soddisfacimento dei criteri di Leipala e Nevalainen
Stabilità = 10	=>	soddisfacimento dei criteri di Carpenter e W.B.Dowsland

I voti che vanno dal 4 al 10 vengono così stimati: sono attribuiti dei malus per ogni taglio a ghigliottina o buchi presenti nei layouts, mentre sono assegnati dei bonus qualora vengano soddisfatti i criteri di Leipala e Nevalainen. Si osservi che dal punto di vista computazionale, subito dopo l'analisi delle condizioni necessarie, è meglio valutare i criteri di Carpenter e W.B.Dowsland in quanto, se questi risultano soddisfatti, si può evitare l'analisi dei criteri di Leipala e Nevalainen dato che risulterebbero certamente verificati.

Si osservi che fino a questo momento si è parlato esclusivamente della stabilità di uno strato su un altro. Se L1 e L2 sono gli strati in questione, alternandoli in generale, si avranno due diversi livelli di stabilità a seconda di quale strato consideriamo come superiore o inferiore.

Indicando con

$$L1suL2 = \text{Livello di stabilità dello strato L1 sullo strato L2} \qquad (1)$$

e

$$L2suL1 = \text{Livello di stabilità dello strato L2 sullo strato L1} \qquad (2)$$

La stabilità complessiva della coppia verrà stimata con la media geometrica delle due combinazioni. In questo modo basterà che uno dei due risultati risulti nullo per inficiare la stabilità complessiva della coppia. Questo è logico in quanto, qualora L1suL2 avesse stabilità nulla sarebbe inutile che L2suL1 risultasse anche con stabilità 10.

$$Stab_{coppia} = \sqrt{L1suL2 \times L2suL1} \qquad (3)$$

4.6. Determinazione della miglior coppia

Il problema finale della pallettizzazione, si sposta quindi sulla determinazione della miglior coppia. La miglior coppia non deve però essere intesa semplicemente come la coppia più stabile, bensì come la coppia che inserisca il numero massimo di pezzi, soddisfi i criteri di stabilità richiesti, e risulti la più efficiente dal punto di vista delle prese-rilasci e dell'utilizzo della linea di trasmissione. In pratica è molto raro che si riesca ad ottenere una coppia con stabilità 10 realizzata con layouts entrambi a prese minime: di solito occorre un compromesso. Tutto dipende dalle esigenze dell'utente: se è interessato ad un livello di stabilità alto, allora molto probabilmente sarà anche disposto ad impiegare più tempo per il completamento del pallet; al contrario, se l'obiettivo è soltanto quello di evitare l'uso dell'interfalda e si stima che il soddisfacimento dei criteri di Leipala e Nevalainen sia sufficiente a questo scopo, sarà inutile andare a cercare configurazioni più stabili e si sceglierà la coppia che raggiunge quel determinato livello di stabilità e impiega meno prese totali. Per determinare tale coppia abbiamo bisogno di una funzione che, dato un certo strato, riesca a determinare il miglior layout da associargli secondo la totalità dei criteri suddetti. Definiamo *gemello* tale layout da associare. Conseguentemente la funzione che associa ad un layout il suo gemello sarà denominata funzione *'CercaGemello'*. Si osservi che per implementare una simile funzione basterebbe valutare tutte le coppie formate dal layout in questione e tutti gli altri e tenere la migliore. Tuttavia nell'analisi delle coppie, non possiamo valutare soltanto le combinazioni tra tutti i layouts prodotti dall'algoritmo GeneraLayers, ma dobbiamo inserire nel computo anche tutte le loro eventuali trasformazioni. Per trasformazione di un layout si intendono:

1. Rotazione 180°
2. simmetrizzazione rispetto all'asse X
3. simmetrizzazione rispetto all'asse Y

Non deve essere esclusa neppure la valutazione della coppia formata dallo strato e da una sua trasformazione: anzi, in molti casi, specie se il layout è quello prodotto da un algoritmo a 4-blocchi,

può essere una buona soluzione dal punto di vista della stabilità. Oltretutto, Carpenter e W.B.Dowsland hanno dimostrato che, nel caso si operi con un layout e un suo derivato, sarà sufficiente verificare la stabilità di uno sull'altro e non anche del viceversa, in quanto questa ultima ne risulterà una diretta conseguenza [23]. In questo modo, se N è il numero dei layouts prodotti da GeneraLayers, per ognuno di essi si dovrebbero valutare 4xN coppie. In totale alla fine ci troveremmo ad aver valutato $4xN^2$ coppie. E' evidente che, data la complessità delle valutazioni, al crescere di N, i tempi computazionali finirebbero per non essere indifferenti. Si può evitare di dover valutare il layout in questione con tutti gli altri se si riesce a trovare la soluzione migliore per prima. Così facendo la lista dei layouts a disposizione verrà opportunamente riordinata in base al numero di prese e al coefficiente di utilizzo della linea di trasmissione associato. In questo modo la funzione CercaGemello non dovrà fare altro che prendere un layout e scorrere i layouts della lista partendo dalla testa e valutarne la stabilità di volta in volta: la prima coppia che soddisfi il livello di stabilità richiesto risulterà anche la miglior soluzione dal punto di vista delle prese e di conseguenza si potrà arrestare la ricerca. Una volta associato ad ogni layout il proprio gemello sarà facile fare una comparazione tra le varie coppie e stabilire quella più congeniale per la pallettizzazione.

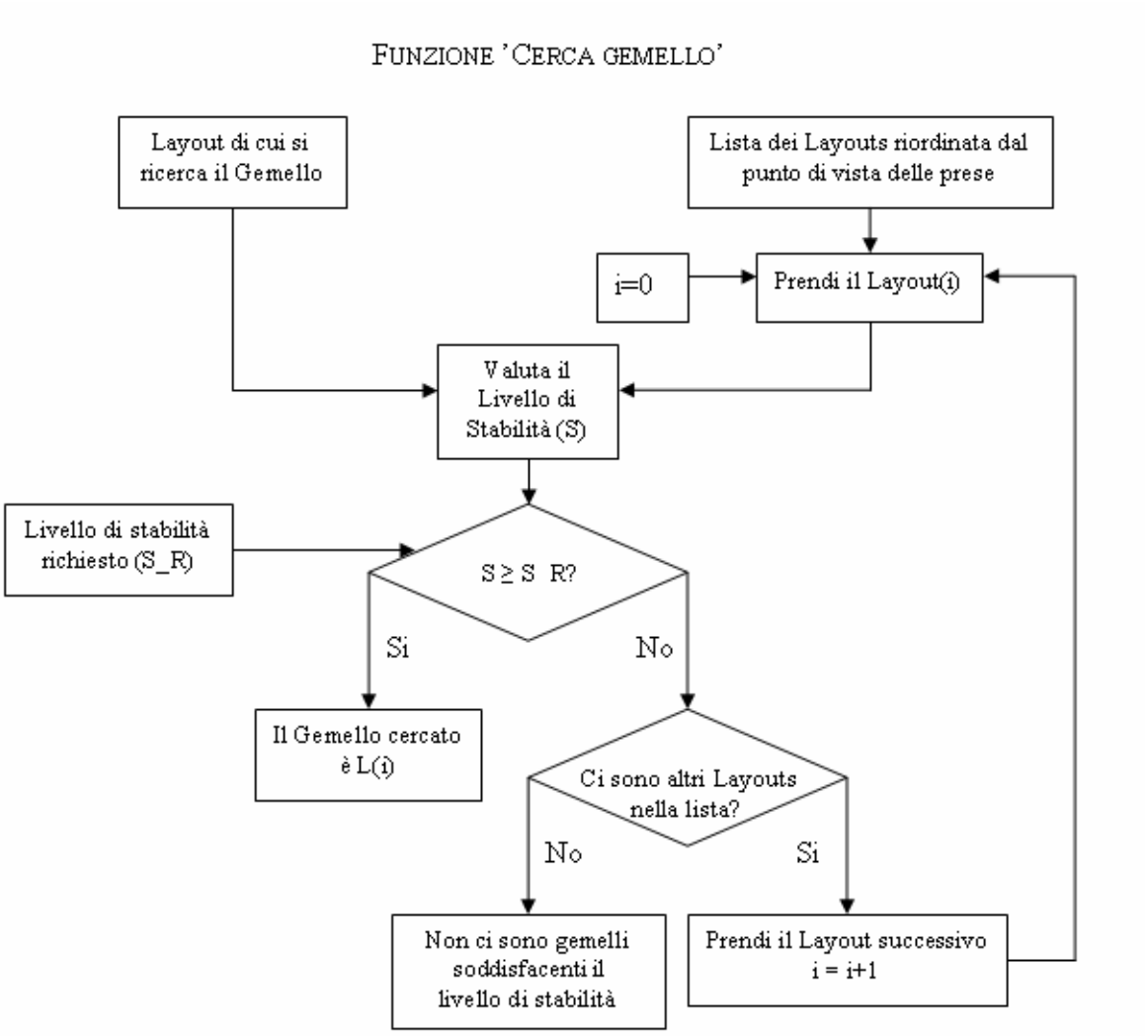


Figura 4.9: ***Funzione 'CercaGemello'***

Capitolo 5 : Descrizione del programma PALLETTIZZA

Seguendo le idee sviluppate nei precedenti capitoli è stato progettato un programma in Visual C++ 6.0 che cerca di risolvere in modo efficiente il problema della pallettizzazione. Tale programma è stato denominato 'PALLETTIZZA'. Per facilitare l'interazione con l'utente esterno il programma è impostato su cinque finestre di dialogo che guidano l'utente nel percorso della pallettizzazione: dall'inserimento dei dati iniziali fino alla visualizzazione dei risultati finali.

Ogni finestra assorbe ad una funzione ben precisa che possiamo così riassumere:

Finestra N. 1 : Inserimento dei dati iniziali

Finestra N. 2 : Generazione degli strati in visione pezzi

Finestra N. 3 : Generazione degli strati in visione prese

Finestra N. 4 : Generazione delle coppie secondo i criteri di stabilità

Finestra N. 5 : Visualizzazione dei risultati finali

All'interno di ogni finestra l'utente può interagire con il programma PALLETTIZZA stabilendo di volta in volta come gestire i dati a disposizione e visualizzando i risultati parziali ottenuti. Per favorire eventuali ripensamenti ogni finestra è dotata di un tasto 'BACK' che permette all'utente di tornare indietro al dialogo precedente e rivedere le proprie decisioni.

Passiamo in rassegna ogni singola finestra:

5.1. Inserimento dei dati iniziali

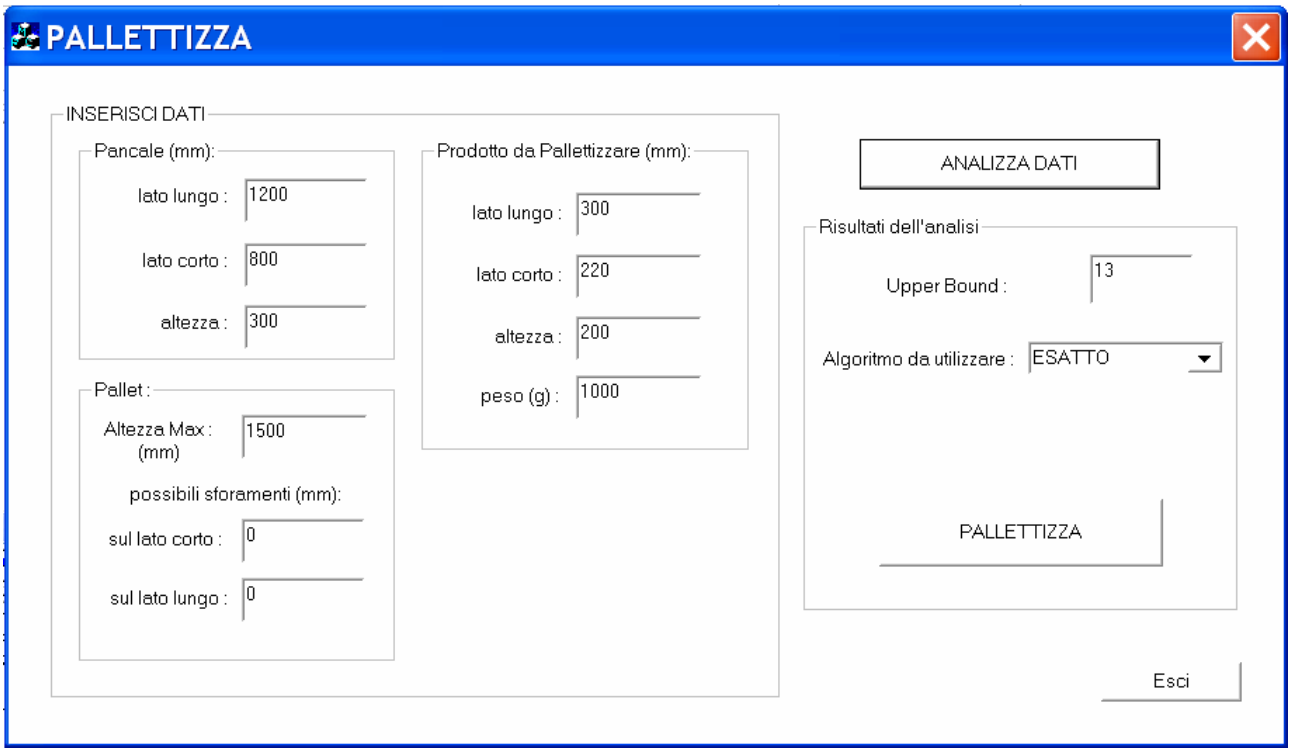


Figura 5.1: Finestra per l'inserimento dei dati iniziali

Questa è la prima finestra che appare all'utente: oltre agli ovvi inserimenti dei dati relativi alle dimensioni del prodotto e del pancale, viene offerta la possibilità di inserire un eventuale sforamento possibile sui lati del pallet. Questa soluzione è spesso usata nei sistemi di pallettizzazione reali in quanto spesso uno sforamento di pochi centimetri produce un inserimento di un pezzo in più per strato.

Una volta definiti i dati iniziali e lo sforamento si procede ad una prima analisi mediante il bottone 'ANALIZZA DATI'. A questo punto diviene accessibile anche il riquadro 'Risultati dell'analisi' nel quale il programma fornisce una prima valutazione dell'UpperBound del problema e consiglia l'algoritmo da utilizzare per la generazione degli strati. Come già spiegato, PALLETTIZZA ha al suo interno due algoritmi, uno esatto e uno euristico, per la pallettizzazione dei prodotti a base rettangolare: l'utente ha la possibilità di scegliere l'algoritmo che ritiene più opportuno cambiando l'impostazione su 'Algoritmo da utilizzare'. All'interno del programma è stato inserito anche un

semplice algoritmo di pallettizzazione per i prodotti a base quadrata che rappresentano un caso degenere dei prodotti a base rettangolare. In quest'ultimo caso l'utente non è abilitato ad alcun tipo di scelta.

Mediante il bottone 'Pallettizza' si accede alla finestra successiva.

5.2. *Generazione degli strati in visione pezzi*

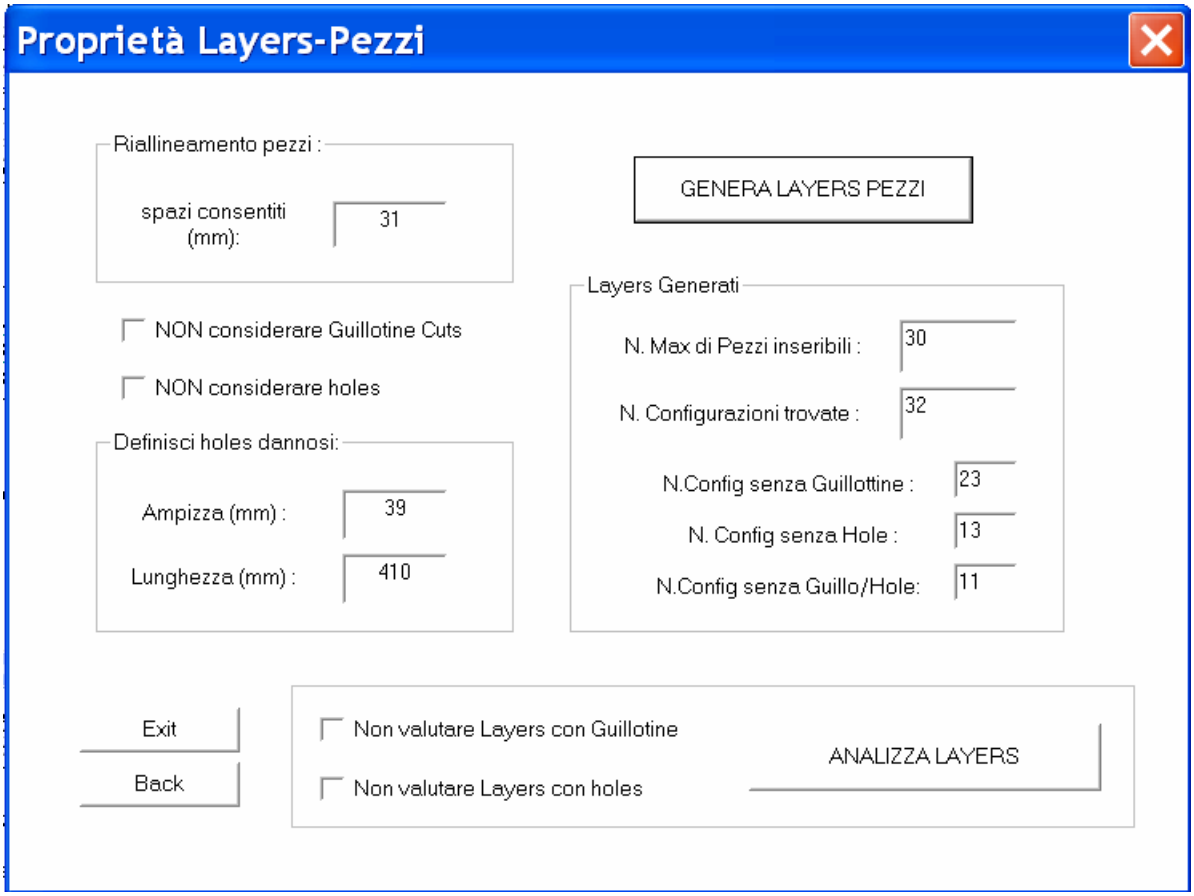


Figura 5.2: Finestra per la generazione degli strati in formato pezzi

Con la seconda finestra viene definita la composizione degli strati in formato pezzi, ossia viene creata la lista degli strati che inseriscono il numero massimo di pezzi. A questo punto del programma ogni strato è caratterizzato dalla collocazione e dall'orientazione dei pezzi che lo compongono. In questa fase l'utente può intervenire direttamente sulla ricostruzione dei layouts mediante il parametro 'spazi consentiti' nel riquadro 'Riallineamento pezzi'. Questo parametro influisce direttamente sulla funzione *Perimetrizza*, permettendo di stabilire il riallineamento dei pezzi che possono traslare lungo una direzione. Vedremo dai test, come il suo valore possa influenzare le future analisi sia sulla stabilità che sul numero di prese necessario per la ricostruzione fisica dello strato. Per velocizzare i tempi, durante la composizione degli strati, vengono anche tratte delle informazioni che saranno successivamente utili per l'analisi della stabilità. In questo

modo, a meno che l'utente non deselezioni l'opzione in quanto non interessato, per ogni strato saranno stabiliti anche gli eventuali tagli a ghigliottina presenti e gli eventuali buchi di particolari dimensioni. Tali dimensioni dei buchi devono essere stabilite dall'utente. Se viene attivata l'opzione 'NON considerare holes' anche i parametri relativi ai buchi diverranno inaccessibili per l'utente in quanto perderanno di significato. Mediante il bottone 'GENERA LAYERS PEZZI' si attiverà il riquadro 'Layers Generati' in cui vengono forniti i risultati ottenuti fino a quel punto. Si osservi che, nel caso l'utente non sia effettivamente interessato all'analisi dei tagli a ghigliottina o dei buchi, è sempre consigliabile la loro disattivazione in modo da evitare tutti i calcoli computazionali ad essi relativi. Al contrario, se l'utente vuole assolutamente evitare gli strati con tagli a ghigliottina o con buchi da lui definiti, ha la possibilità di escluderli dalle successive analisi selezionando le opzioni 'NON valutare layers con Guillotine' e 'NON valutare layers con holes'. In questo modo specie quando sono generate un eccessivo numero di soluzioni, si rende più veloce la computazione per le analisi successive. Per poter escludere i layers occorre che ne rimangano altri a disposizione altrimenti l'opzione non risulta attivata. Una volta fatte queste scelte, azionando 'ANALIZZA LAYERS' si passa alla finestra per la determinazione delle prese-rilasci.

5.3. Generazione degli strati in visione prese

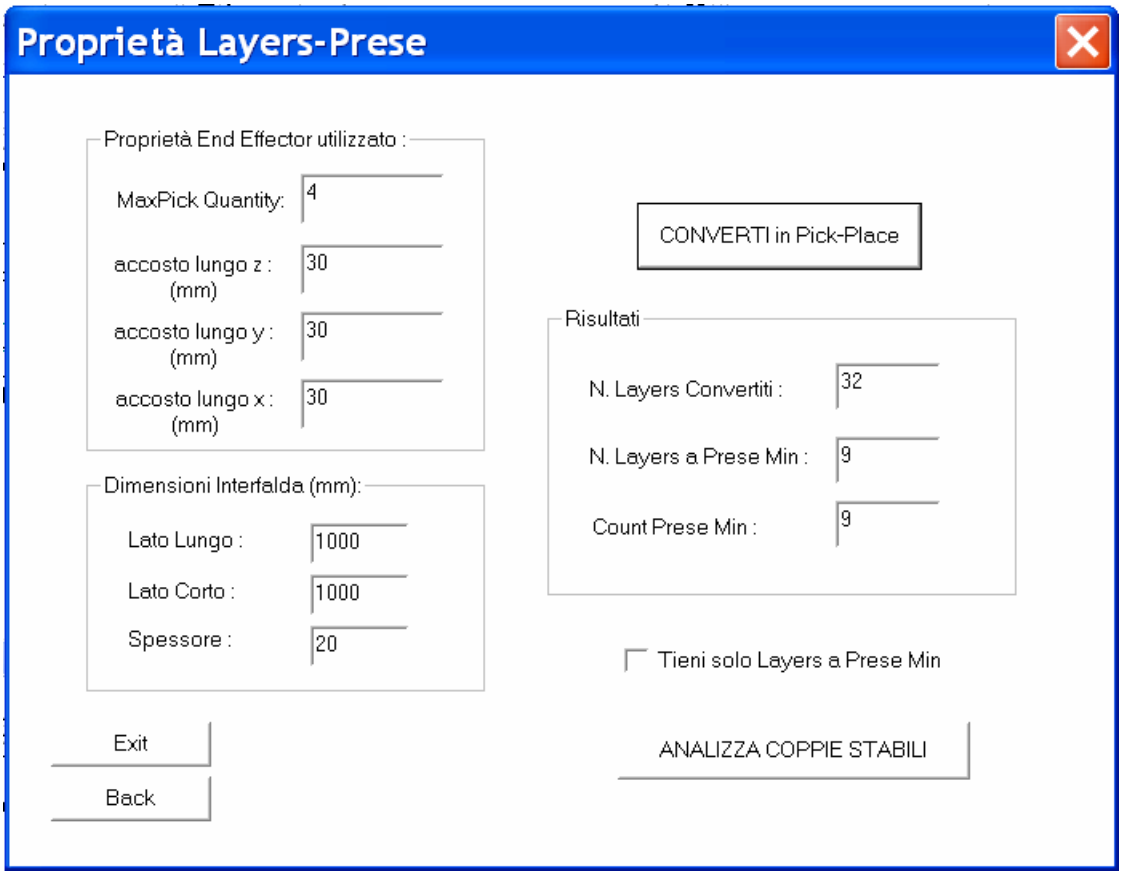


Figura 5.3: Finestra per la conversione in prese-rilasci

Per la conversione degli strati in formato prese-rilasci, l'utente deve fornire le informazioni relative al robot utilizzato per la pallettizzazione, ovvero quanti prodotti è in grado di trasportare contemporaneamente e gli eventuali accosti necessari. Vengono inseriti anche i dati relativi alle dimensioni dell'interfalda che comunque di default è assunta grande come il pancale. Azionando 'CONVERTI IN PICK-PLACE' si attiva il riquadro 'Risultati' in cui vengono evidenziati il numero di layers diversi che risultano ricomponibili con il numero minimo di prese, e appunto il numero minime di prese necessarie a ricomporli. Se l'utente è poco interessato ad una stabilità raffinata del sistema e vuole invece puntare sulla velocità di realizzazione del pallet completo, può decidere di tenere solo gli strati a prese minime attivando l'opzione 'Tieni solo Layers a Prese Min'.

Con 'ANALIZZA COPPIE STABILI' si passa all'analisi della stabilità.

5.4. Generazione delle coppie secondo i criteri di stabilità

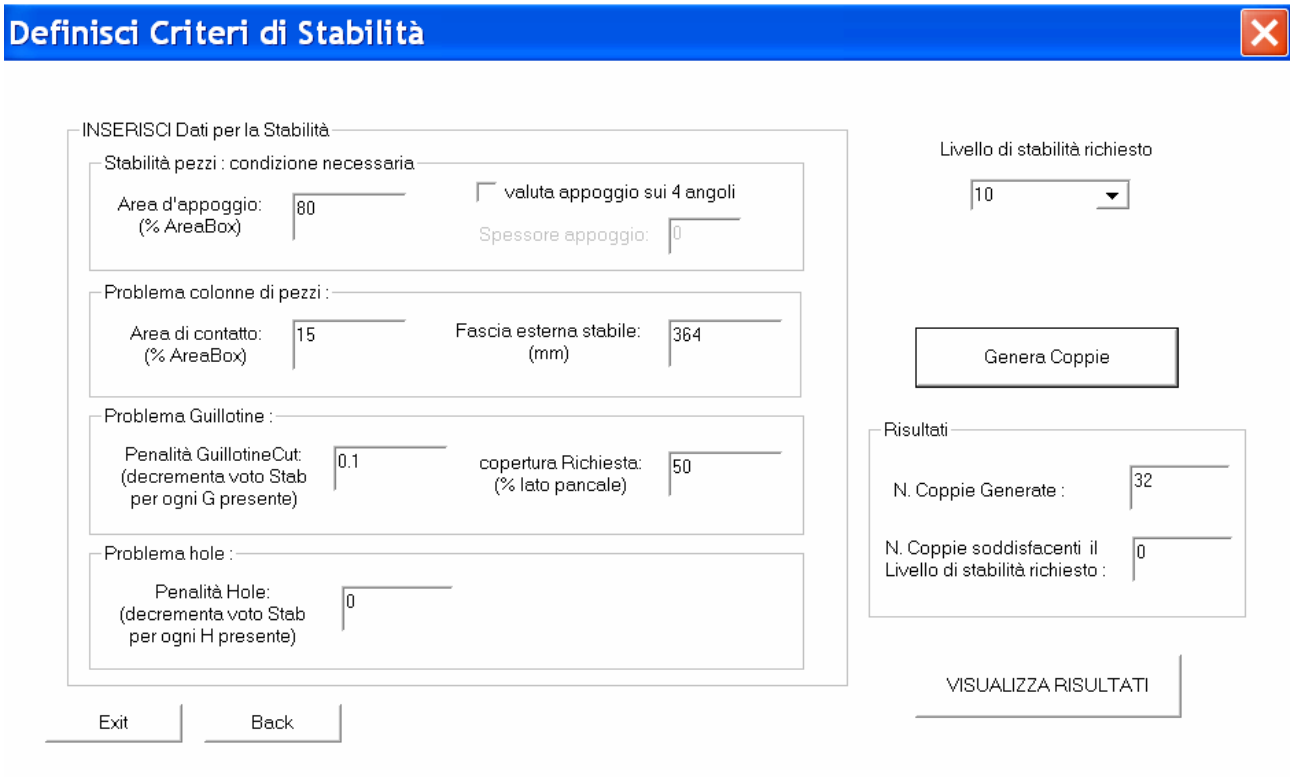


Figura 5.4: Finestra per la generazione delle coppie

In questa finestra l'utente deve definire i coefficienti necessari per l'analisi della stabilità: la condizione necessaria di default è stabilita in un contatto d'appoggio per almeno l'80% della superficie di base del prodotto, oppure nell'appoggino di tutti e quattro gli angoli del prodotto su una porzione di superficie di contatto con spessore pari al 20% del lato corto del prodotto. L'utente ha la possibilità di variare queste percentuali a seconda di come la ritiene più opportune e di disattivare o meno la condizione di corpi rigidi. Una volta passato il test sulle condizioni necessarie, il programma passerà ad analizzare le condizioni opzionali in base ai parametri forniti dall'esterno. Anche in questo caso sono assegnati dei valori di default che comunque l'utente può cambiare come vuole. La superficie di contatto minima, per stabilire se un pezzo appoggia su un altro, è stabilita al 15% della superficie di base del prodotto. La fascia esterna stabile richiesta è stimata con la quantità W_box+L_box, in modo che verso l'esterno del pallet non si formino colonne di pezzi. Viene attribuito 0.1 punti di penalità per ogni taglio a ghigliottina presente negli starti, e viene richiesta

una copertura dei tagli pari al 50% della loro lunghezza. Di defualt non sono introdotte penalità sugli eventuali buchi.

Prima di avviare l'analisi delle coppie, l'utente da 'Livello di stabilità richiesto', può stabilire un livello di stabilità che ritiene sufficiente ai propri scopi. Anche questo è un parametro molto importante perché andrà ad influire direttamente sulle coppie generate. Se ad esempio si ritiene che una stabilita secondo i criteri di Leipala e Nevalainen vada bene, sarà sufficiente fissare un livello di stabilità pari a 8. In questo modo si semplifica la ricerca computazionale del programma che produrrà soluzioni più interessanti dal punto di vista delle prese.

Una volta premuto il bottone 'Genera Coppie' verrà attivato il riquadro dei 'Risultati' in cui viene specificato il numero di coppie analizzate e tra queste, quelle che soddisfano il livello di stabilità richiesto. Può accadere che il livello di stabilità richiesto sia stato troppo alto, e di conseguenza non vengano trovate coppie soddisfacenti tale livello. Andando avanti col tasto 'VISUALIZZA RISULTATI', il programma visualizzerà comunque le migliori soluzioni trovate dal punto di vista della stabilità.

5.5. Visualizzazione dei risultati finali

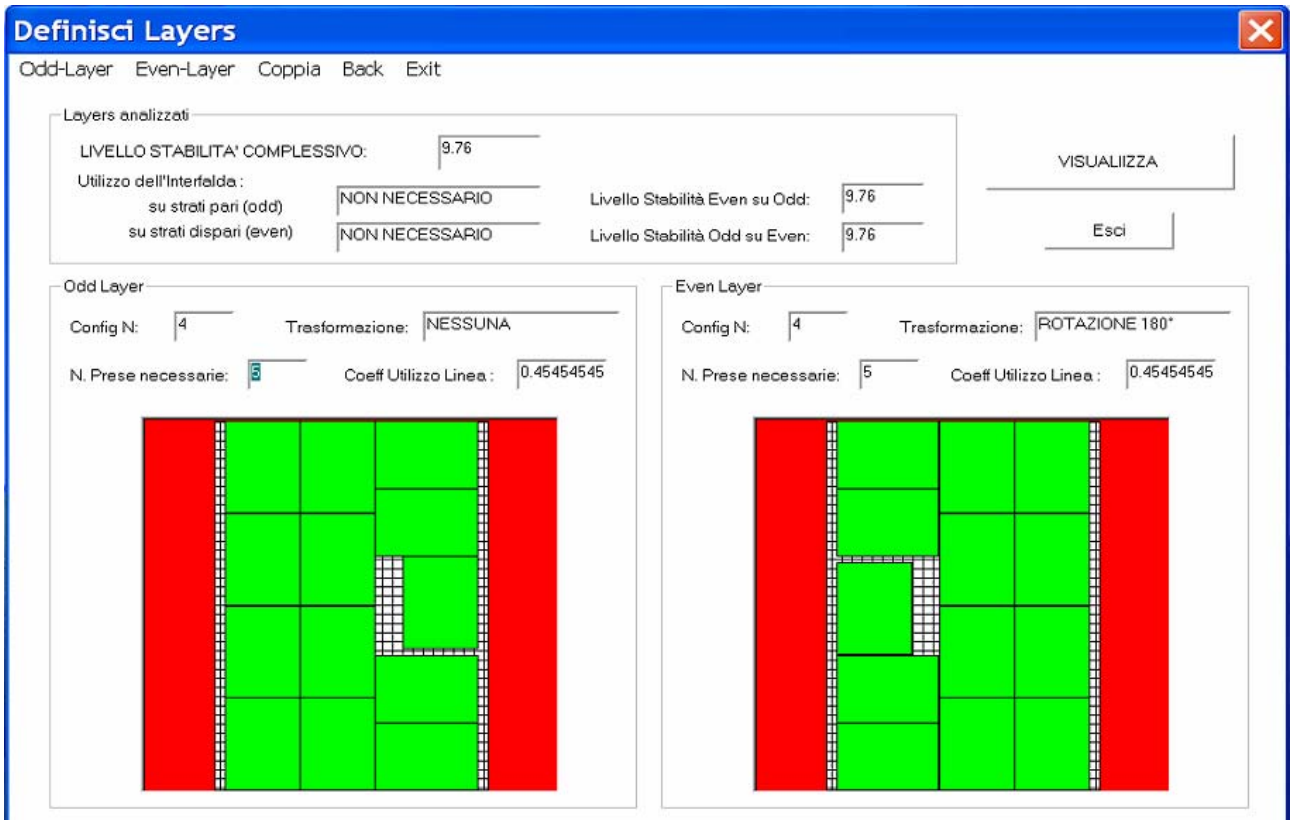


Figura 5.5: Finestra per visualizzazione dei risultati finali

L'ultima finestra visualizza in primo piano la coppia che secondo il programma risulti essere la più efficiente secondo i parametri forniti. In basso vengono visualizzati, in formato 2D, i layouts dei due strati proposti come Odd_Layer e Even_Layer con i relativi parametri informativi sul numero di prese necessarie alla realizzazione e sul coefficiente di utilizzo della linea di trasmissione. In alto viene fornito il livello di stabilità complessivo del sistema, ma sono anche valutati i due livelli di stabilità considerando lo strato pari su quello dispari e viceversa. Tramite questi due parametri il programma stabilisce anche l'eventuale utilizzo o meno dell'interfalda.

La soluzione che appare per prima è quella ritenuta migliore dal programma PALLETTIZZA. Tuttavia durante l'elaborazione ad ogni layout con numero massimo di pezzi è stato associato un corrispondente gemello. Le coppie così generate sono state riordinate in ordine di preferenza. Questa finestra è stata fornita di un ulteriore menù posto in alto tramite il quale l'utente può

rielaborare le soluzioni generate. Tramite il menu è possibile ad esempio, scorrere tutte le coppie generate, apportare delle modifiche ai singoli layout, cambiare sia lo strato pari che quello dispari per poi rivalutarne il gemello da associare. Le trasformazioni possibili sul singolo layer sono la rotazione di 180°, il simmetrico rispetto all'asse 'X' e il simmetrico rispetto all'asse 'Y'. Dopo aver apportato le modifiche ad un singolo layout, richiamando la funzione CercaGemello per quello strato, il programma cercherà di associargli il miglior layer tra quelli in memoria secondo i parametri forniti.

L'inserimento di questo menù finale mira ad offrire un livello di interazione finale tra programma e utente: dato che il risultato proposto è di natura euristica non è garantito che esso sia effettivamente l'ottimo. Di conseguenza un utente esperto potrebbe gradire di visualizzare più soluzioni in ordine di preferenza ed eventualmente anche di interagire con esse. Il software realizzato cerca appunto di soddisfare queste esigenze, fornendo all'utente la possibilità di interagire direttamente col programma durante la fase di elaborazione delle soluzioni, e poi alla fine, permettendo la gestione diretta delle soluzioni finali per ulteriori manipolazioni esterne.

L'ultimo bottone di questa finestra che non è ancora stato descritto, è il tasto 'VISUALIZZA' che serve per collegare l'uscita del programma PALLETTIZZA con un software di visualizzazione fornito dalla ditta '*Scienza Machinale*'. Tramite questo software è possibile vedere l'implementazione in 3D della pallettizzazione finale adottata dall'utente.

5.6. Realizzazione della BNF (Backus Nauer Form)

La soluzione finale prodotta dall'utente tramite il programma PALLETTIZZA può essere tradotta in una BNF e passata ad un software di visualizzazione 3D in grado di mostrare il pallet completo e gli eventuali movimenti del robot necessari per assemblarlo. Nella BNF saranno quindi identificate le dimensioni del pancale, del prodotto, il numero di strati che dovranno essere sovrapposti, le eventuali interfalde, e la descrizione dello strato pari e di quello dispari con tutte le relative prese e rilasci, accosti e rotazioni.

Esempio di BNF prodotta per l'istanza [1200,800,300,220] con tutti i valori de default

```
SCHEME_NAME LAYOUT

BEGIN_PALLET_DESCRIPTION
SIZE 800 1200 300
END_PALLET_DESCRIPTION

BEGIN_SHEET_DESCRIPTION
SIZE 800 1200 20
END_SHEET_DESCRIPTION

BEGIN_PRODUCT_DESCRIPTIO
N
SIZE 220 300 200
WEIGHT 1000
END_PRODUCT_DESCRIPTION

LAYERS_COUNT 7

BEGIN_ODD_LAYER
BEGIN_LAYER_DESCRIPTION
BELOW SHEET 0
BEGIN_PICKPLACE

PICK_QUANTITY 2
PICK_ROTATION  0
PLACE_OFFSET 0 0 30
PLACE_POSITION 620 220 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 4
PICK_ROTATION  1
PLACE_OFFSET -30 0 30
PLACE_POSITION 360 600 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 1
PICK_ROTATION  1
PLACE_OFFSET 30 30 30
PLACE_POSITION 660 590 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 4
PICK_ROTATION  1
PLACE_OFFSET -30 0 30
PLACE_POSITION 140 600 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 2
PICK_ROTATION  0
PLACE_OFFSET 30 30 30
PLACE_POSITION 620 980 0
PLACE_ROTATION 1

END_PICKPLACE

END_LAYER_DESCRIPTION
END_ODD_LAYER

BEGIN_EVEN_LAYER
BEGIN_LAYER_DESCRIPTION
BELOW SHEET 0
BEGIN_PICKPLACE

PICK_QUANTITY 2
PICK_ROTATION  0
PLACE_OFFSET 0 0 30
PLACE_POSITION 180 980 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 4
PICK_ROTATION  1
PLACE_OFFSET 30 0 30
PLACE_POSITION 440 600 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 1
PICK_ROTATION  1
PLACE_OFFSET -30 -30 30
PLACE_POSITION 140 610 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 4
PICK_ROTATION  1
PLACE_OFFSET 30 0 30
PLACE_POSITION 660 600 0
PLACE_ROTATION 1

END_PICKPLACE

BEGIN_PICKPLACE

PICK_QUANTITY 2
PICK_ROTATION  0
PLACE_OFFSET -30 -30 30
PLACE_POSITION 180 220 0
PLACE_ROTATION 1

END_PICKPLACE

END_LAYER_DESCRIPTION
END_EVEN_LAYE
```

Capitolo 6 : Test

Al fine di verificare l'efficacia del programma PALLETTIZZA e di puntualizzare alcuni aspetti tipici del processo della pallettizzazione, sono stati svolti alcuni test su delle particolari istanze di prova. Queste istanze sono state per lo più riprese da precedenti studi, o sono state dettate dall'esperienza diretta dell'azienda *Scienza Machinale* per la quale è stato svolto il lavoro. Tali prove possono servire per valutare i vari parametri di default da inserire all'interno del programma PALLETTIZZA. Ai fini di realizzare un prodotto commerciale è opportuno valutare anche il rapporto con l'eventuale cliente/utente. Un buon programma dovrebbe riuscire a soddisfare le esigenze contrapposte delle più svariate persone che ne usufruiranno. In questa casistica rientrano sia le persone poco esperte, che ambiscono soltanto ad arrivare ad un risultato piuttosto velocemente, senza doversi preoccupare di altre questioni, sia quelle più esperte, che potrebbero invece sentirsi limitate da un software che fornisca solo una soluzione nuda e cruda senza la possibilità di una effettiva interazione. Da questo punto di vista, i valori di default devono essere adeguatamente studiati proprio per riuscire a soddisfare tutti i generi di utenti: l'utente medio che si avvicini al programma non dovrà far altro che schiacciare i bottoni dell'analisi e vedere la soluzione prodotta. E' quindi importante che i valori impostati, riescano a soddisfare il maggior numero di casistiche possibili.

Per le prove che seguono, ove non specificato diversamente, si è considerato un robot pallettizzatore con un end-effector in grado di afferrare quattro prodotti per volta.

6.1. Algoritmo di ricostruzione

Per prima cosa cerchiamo di testare la bontà o meno dell'algoritmo di ricostruzione degli starti utilizzato nel programma.

6.1.1. Confronto con Algoritmi G4

Gli algoritmi di tipo G4 vengono solitamente considerati delle buone soluzioni per i problemi di piccole e medie dimensioni che sono quelli di nostro interesse. Per confrontare i risultati ottenuti dall'algoritmo da noi scelto, abbiamo ripreso le stesse istanze proposte e analizzate da Terno e Scheithauer nel loro articolo sugli algoritmi di tipo G4 [9].

Cominciamo col considerare l'istanza [560,520,120,50] una cui soluzione ottima è riportata in Figura 6.1:

Figura 6.1: istanza [560,520,120,50]; soluzione trovata dagli algoritmi G4 e da PALLETTIZZA

La soluzione proposta dall'algoritmo G4 del citato articolo, trova pieno accordo con i risultati forniti da PALLETTIZZA: è una soluzione ottima sia dal punto di vista delle prese che da quello della stabilità.

Tuttavia il nostro programma trova anche un'altra soluzione altrettanto efficace mostrata in Figura 6.2:

Figura 6.2: istanza [560,520,120,50]; ulteriore soluzione trovata da PALLETTIZZA

Trovare più soluzioni spesso significa avere una maggiore possibilità di scelta e dunque una maggiore possibilità di arrivare ad una soluzione migliore per il nostro problema.

Consideriamo a tal proposito l'istanza [520,330,90,40]

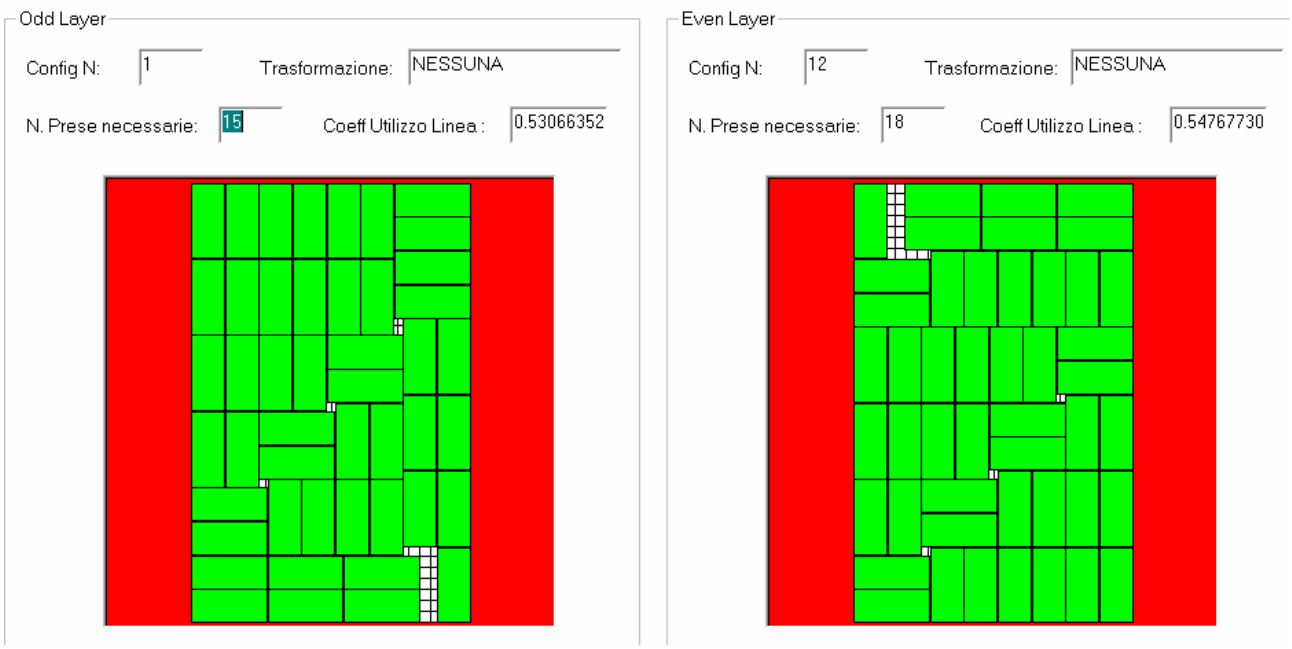


Figura 6.3: istanza [520,330,90,40]; soluzioni trovate da PALLETTIZZA

La soluzione di destra della figura 3 è quella proposta dall'algoritmo G4 sempre del riferimento [9]. Come si può osservare dall'immagine, anche PALLETTIZZA è riuscito a trovare questa soluzione, tuttavia il programma è stato in grado di trovare ben altre 18 soluzioni diverse, tra cui quella illustrata a sinistra. Dal punto di vista del computo delle prese c'è una certa differenza tra le due: utilizzando un robot in grado di depositare 4 prodotti per volta, l'algoritmo da noi proposto troverebbe una soluzione migliore dal punto di vista delle prese-rilasci, in quanto impiegherebbe soltanto 15 movimenti per collocare tutti i prodotti dello strato, contro i 18 movimenti necessari per riprodurre lo schema di destra. Uno dei principali punti deboli degli schemi prodotti dagli algoritmi G4 è inerente alle prese-rilasci: gli schemi che ne derivano sono quasi sempre molto disomogenei dal punto di vista del collocamento dei pezzi. L'intreccio dei prodotti può produrre una buona stabilità, ma risulta penalizzante dal punto di vista dell'efficienza delle prese.

Concludiamo infine con l'istanza [570,440,120,50]

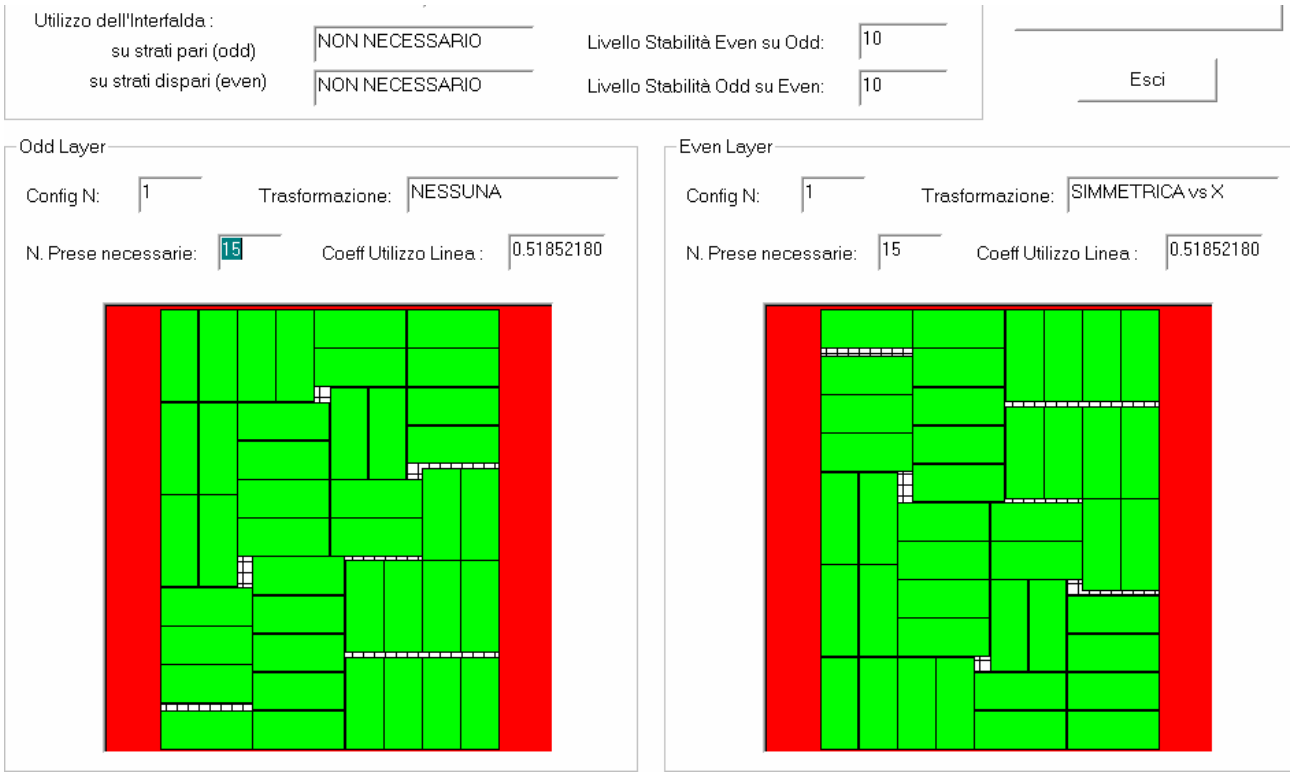


Figura 6.4: istanza [570,440,120,50]; soluzioni trovate da PALLETTIZZA

Stavolta PALLETTIZZA non riesce a trovare la soluzione proposta dall'algoritmo di tipo G4, tuttavia le soluzioni proposte risultano ugualmente affidabili dal punto di vista della stabilità, e al solito in diversi casi, impiegano un minor numero di prese per la ricostruzione dello schema.

In generale possiamo concludere che l'algoritmo proposto è in grado di risolvere le stesse istanze di un euristico di tipo G4: le soluzioni trovate nella maggior parte dei casi, se non uguali, sono molto simili, o addirittura migliori dal punto di vista dell'efficienza delle prese.

6.1.2. Valutazione dell'algoritmo di ricostruzione Euristico a blocchi

Come già detto, il problema degli euristici a blocchi è che non sono molto affidabili dal punto di vista dell'inserimento massimo dei pezzi. Basta considerare una delle precedenti istanze:

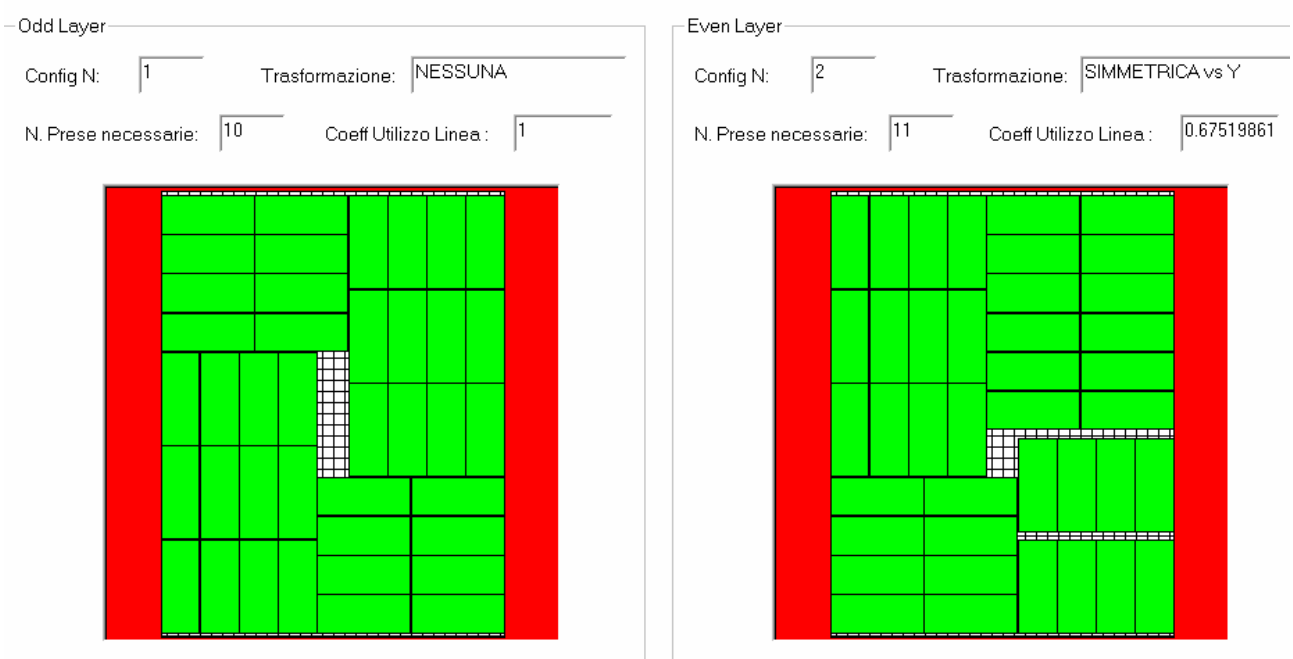
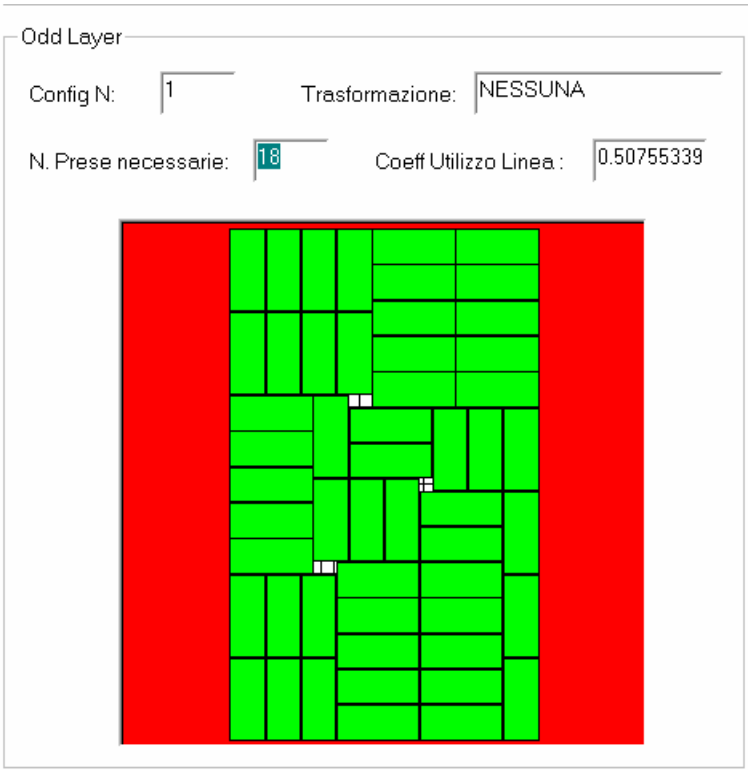
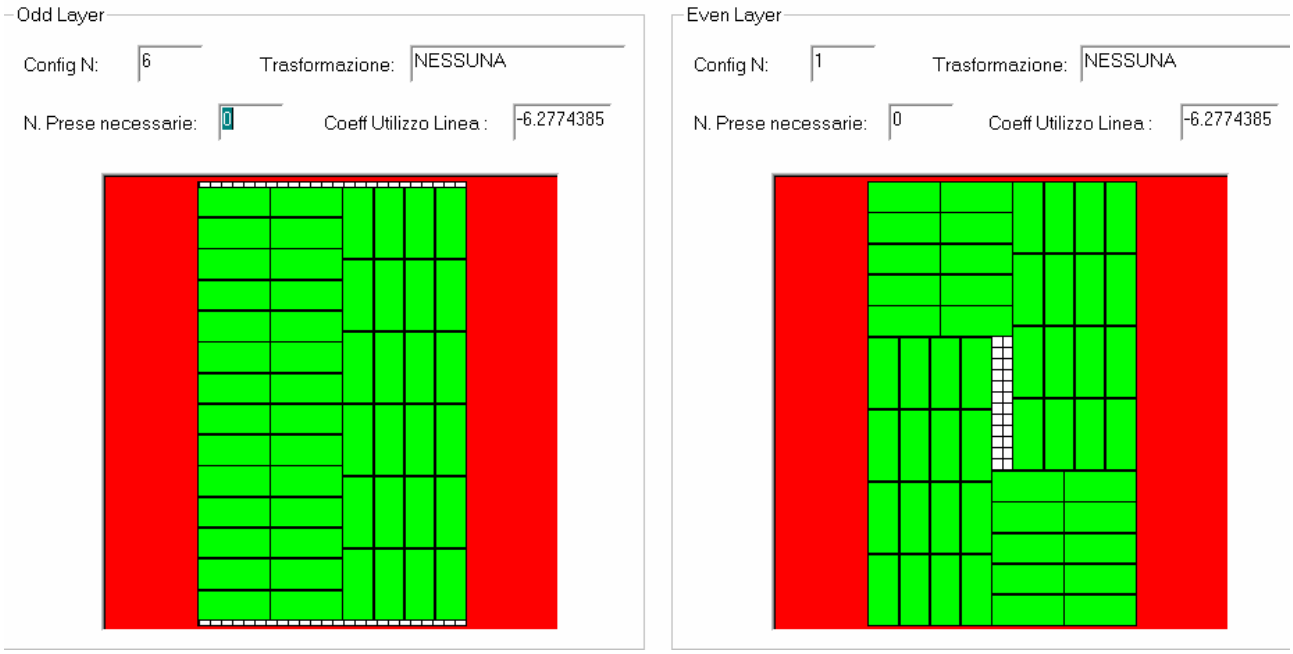


Figura 6.5: istanza [570,440,120,50]; soluzioni euristiche trovate da PALLETTIZZA

L'algoritmo esatto riesce a inserire 41 prodotti per strato: l'euristico a 4 Blocchi si ferma a soli 40. L'istanza [430,260,70,30] è citata in da Morabito e Morales in [10] per testare la bontà di un algoritmo euristico: un simile problema ha un'unica soluzione contenete 53 pezzi. Un semplice euristico a 4 Blocchi non riesce a pallettizzare in modo corretto e si ferma a 52 blocchi inseriti. Inserendo un pezzo in meno vengono a generarsi più soluzioni che sono proposte dall'euristico. In simili circostanze un algoritmo esatto può risultare addirittura veloce dell'euristico in quanto, trovando un'unica soluzione, ottiene un risparmio di tempo nelle analisi successive per l'utilizzo della linea di trasmissione e l'analisi della stabilità.



algoritmo esatto,53 prodotti inseriti, unica soluzione trovata

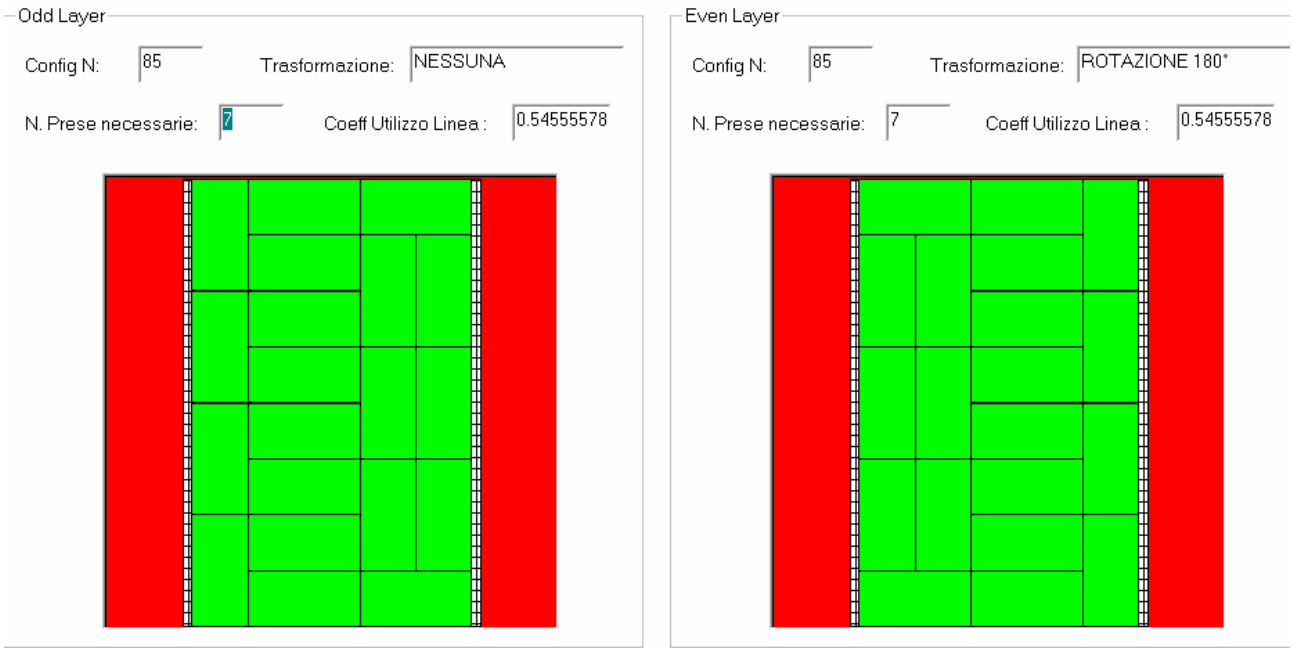


Algoritmo euristico (52 prodotti inseriti) : 2 Soluzioni trovate su 9 soluzioni complessive

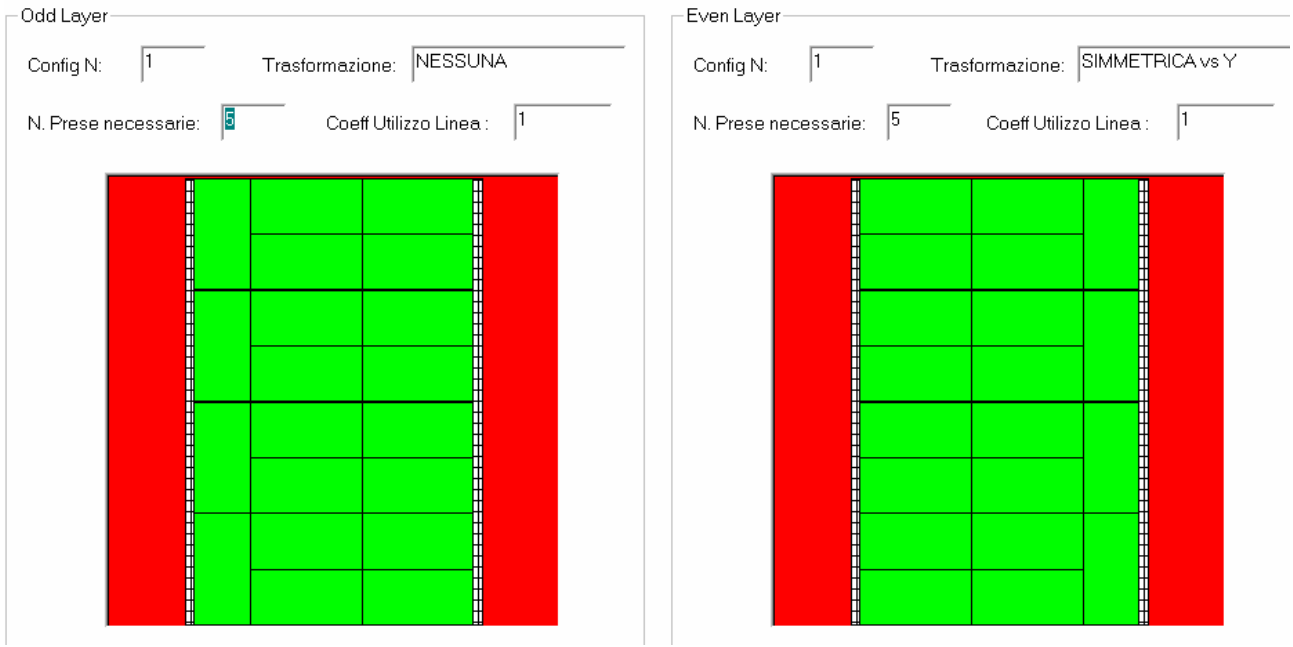
Figura 6.6: istanza (430,260,70,30); soluzione esatta ed euristiche

Questo non significa che l'algoritmo esatto sia sempre da preferire all'euristico. In particolare quando le dimensioni del prodotto sono esattamente multiple l'una dell'altra, l'algoritmo esatto finisce par calcolare tutte le possibili permutazioni con conseguente dilatamento dei tempi di

computazione a causa dell'eccessivo numero di layouts che si generano. Consideriamo ad esempio la seguente istanza [1200,800,300,150]



Istanza (1200,800,300,150) Algoritmo Esatto : 353 soluzioni trovate (tempo di analisi dell'ordine di qualche minuto)



Algoritmo euristico : 9 soluzioni trovate in maniera quasi istantanea

Figura 6.7: istanza (1200,800,300,150); soluzioni esatte ed euristiche

Come si può osservare l'algoritmo esatto ha fornito una soluzione migliore dal punto di vista della stabilità, in quanto sebbene non sia riuscito ad evitare i tagli a ghigliottina sul layout, è riuscito a trovare una sufficiente copertura, grazie ai prodotti che si intrecciano bene. L'algoritmo euristico non è riuscito in questo intento, tuttavia dal punto di vista delle prese-rilasci, la soluzione euristica appare nettamente migliore: i vari strati dell'euristico potranno essere ricostruiti con 5 soli movimenti più un eventuale movimento dovuto all'inserimento dell'interfalda, contro i 7 movimenti per strato per costruire lo schema proposto dall'esatto. In simili casi è bene ricordare che l'utilizzo dell'interfalda non solo ha un costo oggettivo, ma fondamentalmente equivale anche ad una ulteriore presa-rilascio da aggiungere ai tempi di completamento del pallet. In questo caso, nonostante l'aggiunta dell'interfalda la soluzione euristica sarebbe più veloce da ricostruire.

Consideriamo anche la seguente situazione

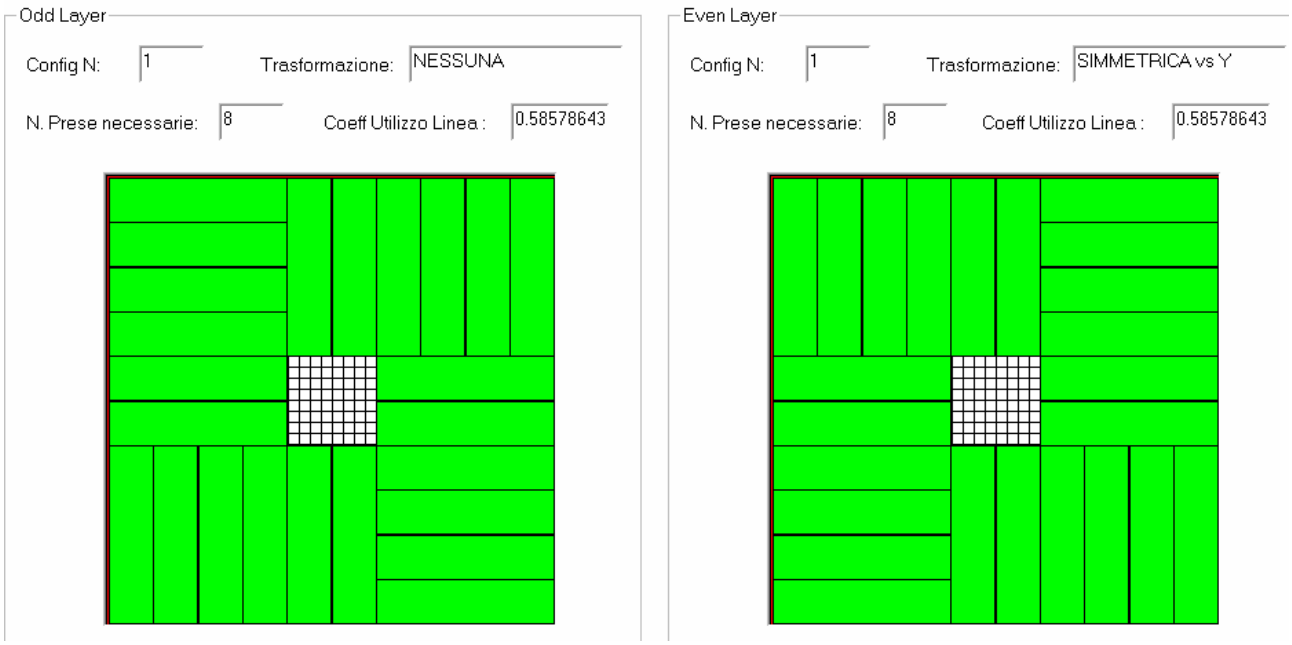


Figura 6.8: istanza (1000,1000,400,100); algoritmo Euristico: 4 Soluzioni trovate

L'algoritmo esatto trova ben 166 soluzioni impiegando vari secondi sia per l'analisi delle prese sia per l'analisi della stabilità.

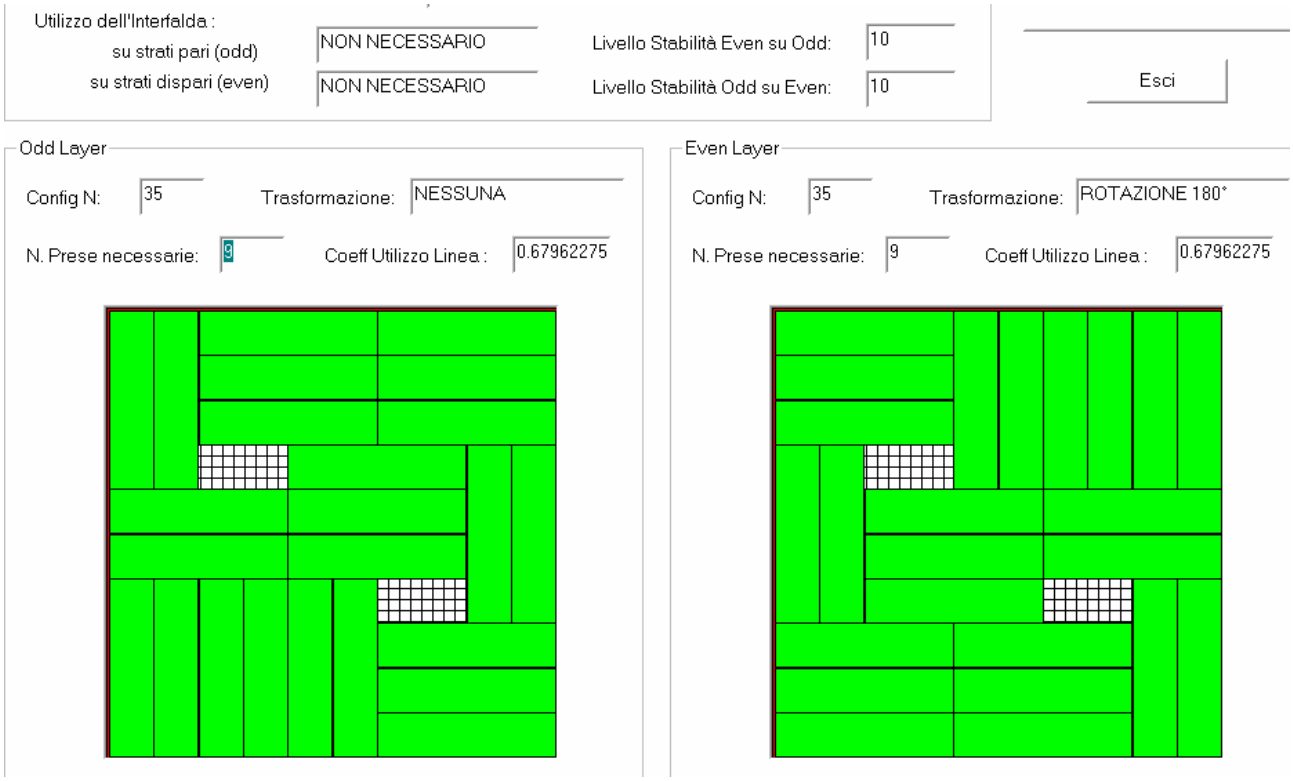


Figura 6.9: istanza (1000,1000,400,100); algoritmo Esatto: 166 Soluzioni trovate

La pazienza in questo caso paga perché la soluzione trovata riesce addirittura ad evitare il problema dei tagli a ghigliottina. Naturalmente tutto ciò si sconta al prezzo di una presa in più per ricostruire ogni singolo strato. Tuttavia considerando anche il collocamento dell'interfalda, possiamo stimare che i tempi di ricostruzione del pallet completo sarebbero pressoché gli stessi, con la differenza che utilizzando lo schema proposto dall'esatto si risparmierebbe l'interfalda.

In definitiva possiamo concludere che se si vuole evitare di utilizzare l'interfalda, è sempre meglio provare ad utilizzare l'algoritmo esatto, in quanto valutando tutte le possibili permutazioni di prodotti, riesce quasi sempre ad intrecciarli meglio. Proprio questo maggiore intreccio però, finisce per causare un aumento del numero di prese. Al contrario, se si è disposti ad utilizzare l'interfalda per cercare di guadagnare sull'efficienza delle prese, allora l'utilizzo dell'euristico può risultare più rapido.

6.2. Problemi nella ricostruzione dei layouts: funzione "Perimetrizza"

L'algoritmo esatto, così come è stato presentato, tende a riempire lo strato partendo da un angolo, nel nostro caso l'angolo alto sinistro, e prosegue l'inserimento allineando via via gli altri prodotti. In questo modo però l'angolo opposto, quello in basso a destra, rischia di risultare quasi sempre mal riempito. D'altro canto, gli angoli del pancale sono delle zone particolarmente delicate per la stabilità che devono essere trattate con riguardo.

Consideriamo il seguente esempio: istanza [1200,800,300,220]

Senza un'adeguata riperimetrizzazione del Layout, l'algoritmo produrrebbe i seguenti schemi:

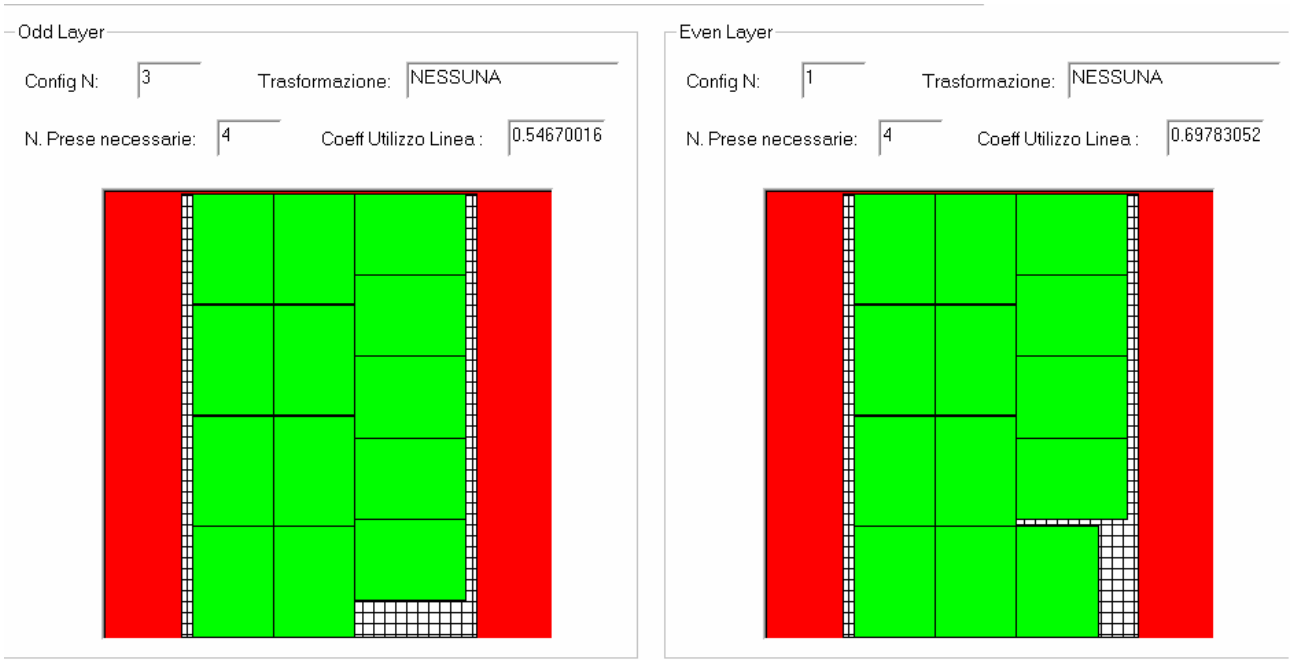


Figura 6.10: istanza [1200,800,300,220]; layouts riprodotti senza la funzione 'perimetrizza'

E' evidente che l'angolo in basso a destra è scarsamente riempito, e ciò renderebbe difficoltoso il sovrapporsi di un qualunque altro tipo di layout diverso dal sottostante. Ovviamente la soluzione di sovrapporre due layers identici è subito da scartare in quanto si verrebbero a creare tutte colonne di prodotti. Per ovviare a questo inconveniente, i layouts prima di essere passati all'analisi delle prese e della stabilità, vengono opportunamente riperimetrizzati lungo le zone più esterne. Per questo è stata implementata la funzione '*perimetrizza*' che ha il compito di valutare le zone vuote lungo i bordi del pallet e valutare se è possibile farvi traslare dei pezzi per riempirle. Il problema è che

possono esistere vari modi di ri-perimetrizzare una certa zona: la Figura 6.11 e la Figura 6.13 ne sono un'esemplificazione. Naturalmente cercare tutte le soluzioni possibili sarebbe quanto meno proibitivo dal punto di vista computazionale, perciò si deve trovare un metodo euristico che consenta di ottenere una buona soluzione nella maggior parte dei casi. Al solito per ottimizzare il nostro sistema, dobbiamo sempre tenere presenti stabilità e efficienza delle prese. Il fatto è che questi due parametri sono spesso in conflitto tra loro: migliorare l'uno può significare peggiorare l'altro, quindi occorre trovare un compromesso. Per quanto riguarda la stabilità, la soluzione migliore sarebbe ripartire tutto lo spazio non ricoperto ridistribuendolo tra i vari pezzi, in quanto così facendo, i prodotti dello strato superiore si troverebbero sempre con dei buchi sottostanti di piccola entità; al contrario dal punto di vista delle prese, sarebbe meglio tenere i pezzi omogenei di uno strato tutti attaccati l'uno all'altro, in modo da poter gestire più prodotti con un'unica presa/rilascio. La soluzione proposta è sempre finalizzata all'utente, il quale, prima della generazione degli strati, nella finestra 'Generazione degli strati in visione pezzi', dovrà settare il valore degli '*spazi consentiti*' per il riallineamento dei prodotti: tale scelta dipenderà dalle esigenze specifiche di maggior stabilità o miglior utilizzo della linea di trasmissione. Abbassando il valore degli spazi consentiti si avrà una ricostruzione finalizzata alla stabilità, mentre alzando tale valore si avrà una costruzione finalizzata all'efficienza delle prese.

Ricostruzione finalizzata alla stabilità: spazi consentiti = 0

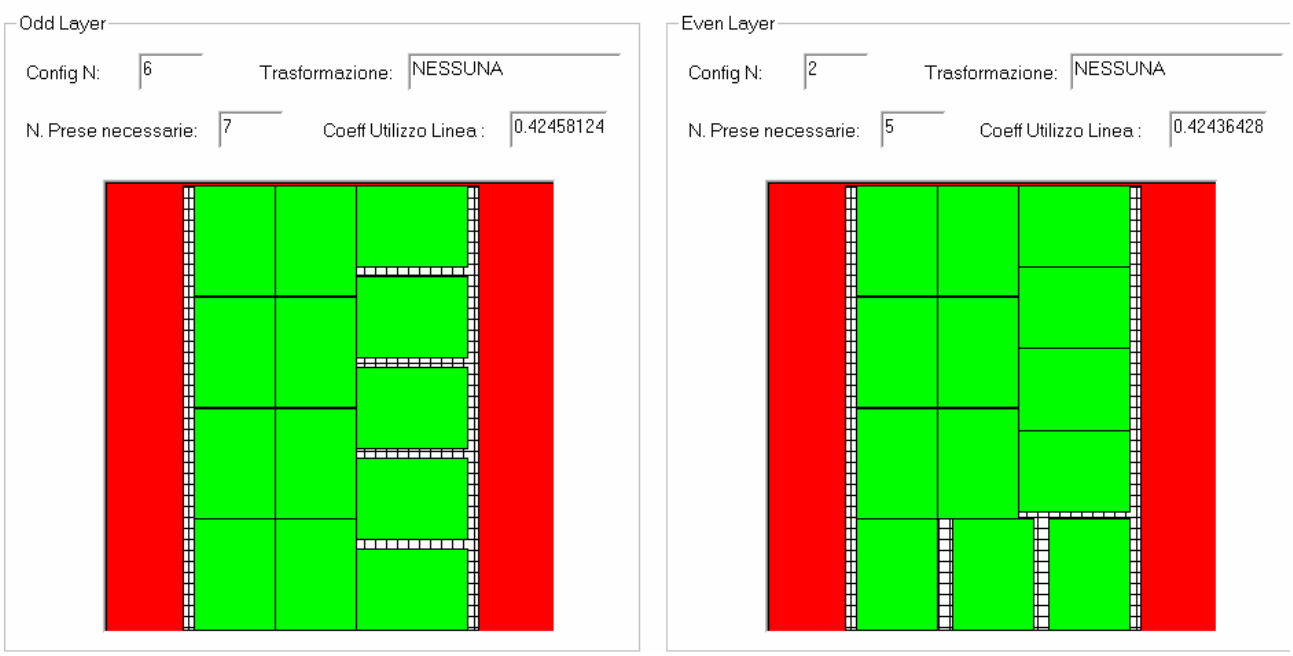


Figura 6.11: Effetto della funzione perimetrizza; ricostruzione finalizzata alla stabilità

Ovviamente da punto di vista delle prese si ha una maggiore dispersione, ma è facile trovare un gemello da associare per la realizzazione del pallet completo:

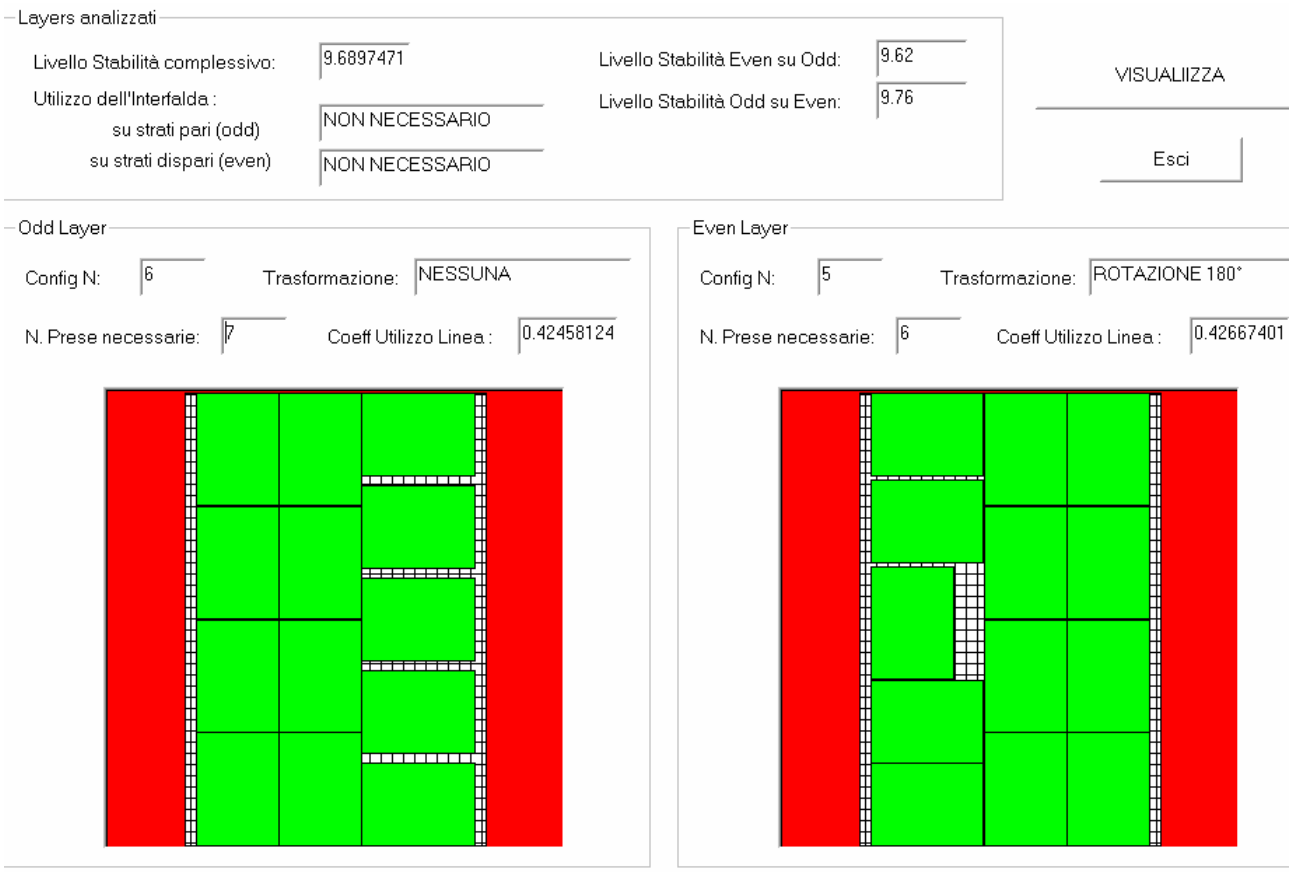


Figura 6.12: soluzione finale proposta con spazi consentiti=0 mm

La coppia qui mostrata non pone problemi di stabilità, ma impiega ben 13 prese per il completamento di entrambi gli strati.

Ricostruzione finalizzata all'efficienza delle prese: spazi consentiti = 300

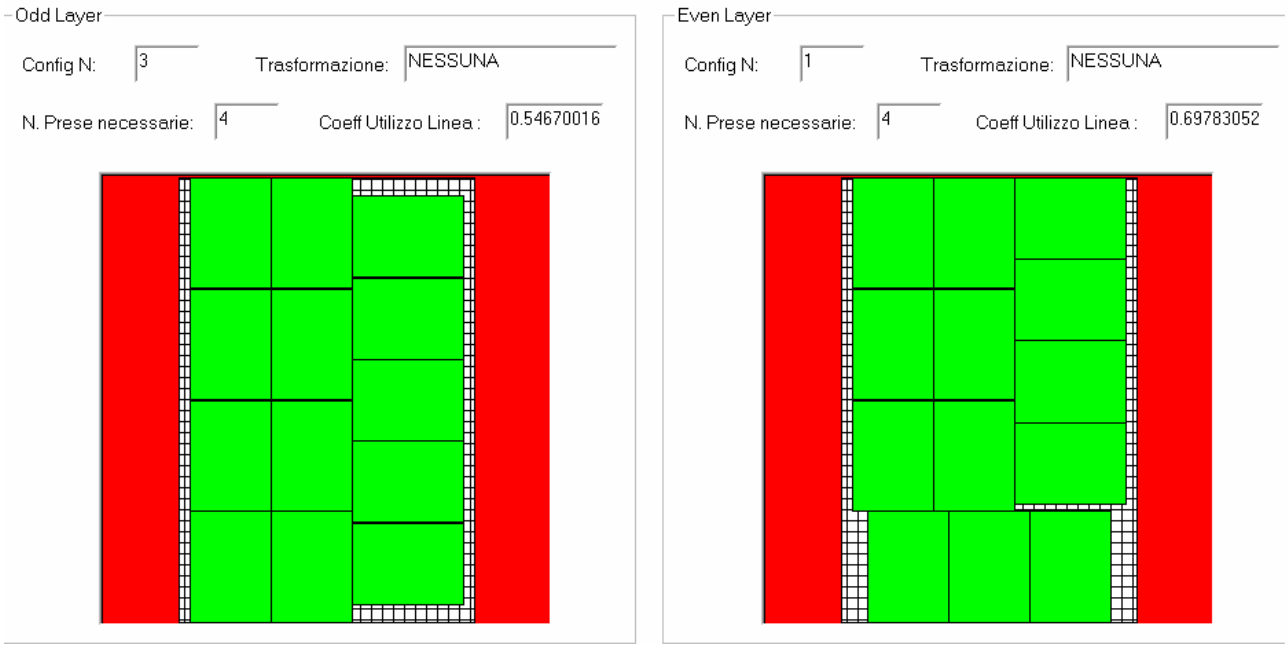


Figura 6.13: Effetto delle funzione perimetrizza; ricostruzione finalizzata alle prese

Lo stesso layout di partenza, stavolta è ricostruito con 4 prese-rilasci, anziché 7, tuttavia stavolta si perde in stabilità:

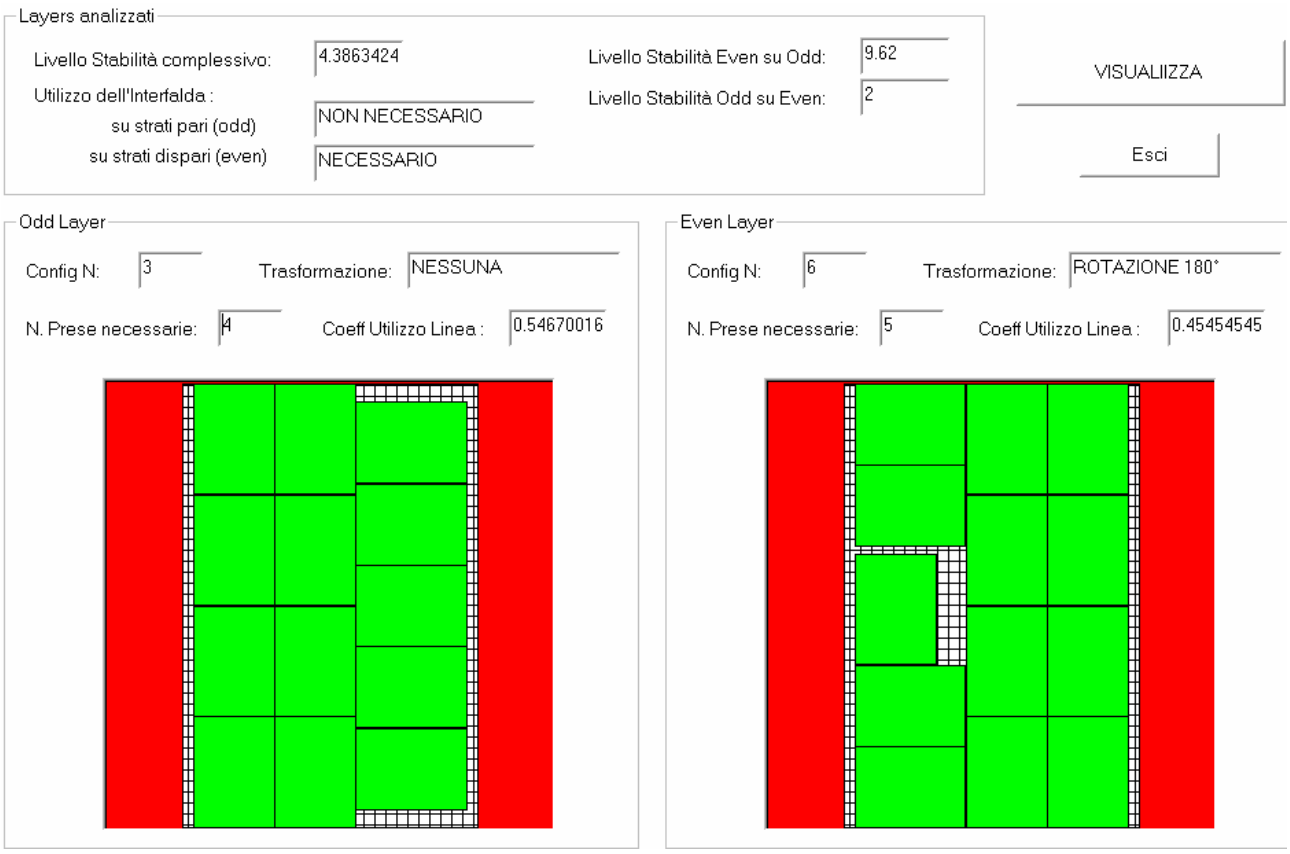


Figura 6.14: soluzione finale proposta con spazi consentiti=300 mm

Il programma propone di associare strato N.3 allo strato N.6 ruotato, ma come si vede dai valori della stabilità, ci sono dei problemi quando si va a sovrapporre il N.6 sul N3, in quanto si creano delle pericolose colonne di pezzi negli angoli alla nostra destra.

Come si può intuire dai ragionamenti prima svolti, solitamente un maggiore dislocamento o intreccio dei prodotti conduce ad una più probabile stabilità, mentre ad un inserimento più ordinato dei prodotti, corrisponde un numero minore di prese-rilasci per ricomporre il layout sul pancale. Come si osserva, prese e stabilità vanno in direzioni opposte: massimizzare l'una può significare minimizzare l'altra. Conseguentemente in molti casi può convenire adottare una soluzione intermedia.

Soluzione intermedia : spazi consentiti = 44

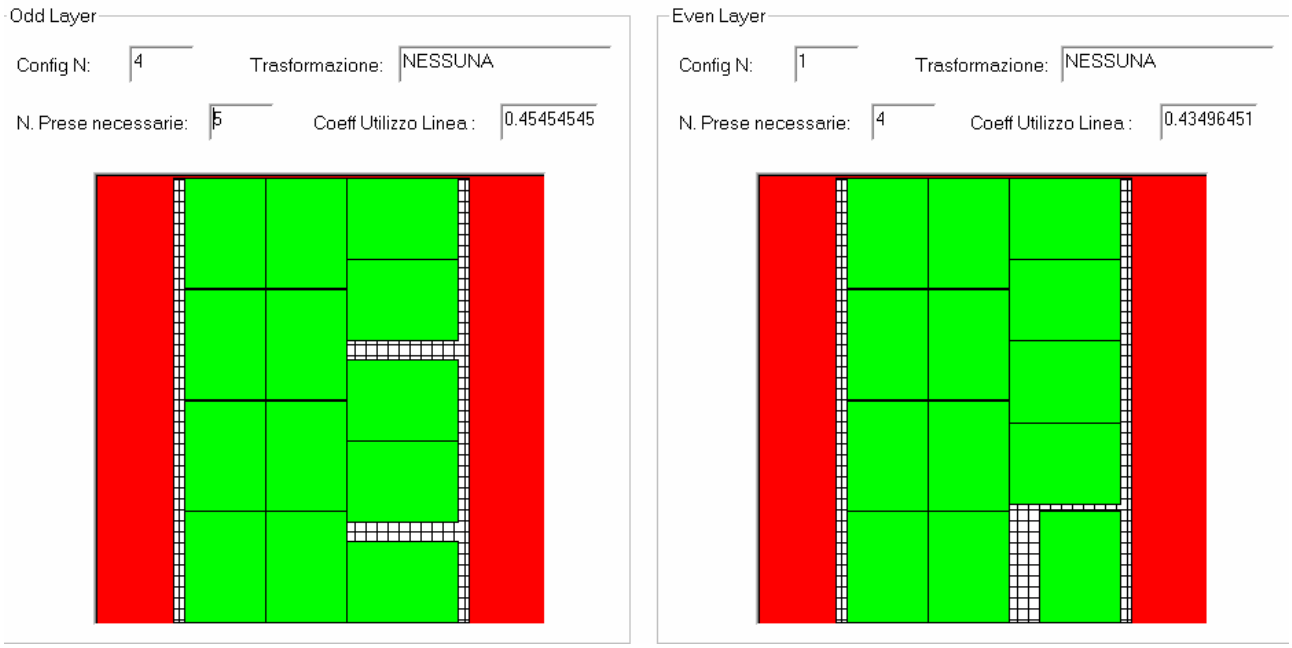


Figura 6.15: soluzione finale proposta con spazi consentiti=44 mm

In questo caso l'utente consente di lasciare un certo spazio tra i pezzi. Così, lo spazio non ricoperto, dove possibile, viene ridistribuito permettendo di mantenere uniti alcuni gruppi di pezzi.

6.3. Lo sforamento

Si consideri l'istanza [1200,800,300,220]: il suo upper bound è 13. Il programma PALLETTIZZA riesce a trovare effettivamente varie soluzioni che inseriscono 13 pezzi e come illustrato in Figura 6.10 anche con un ottima stabilità. Tuttavia basta uno sforamento di 10 mm su entrambi i lati lunghi per ottenere l'inserimento di un ulteriore pezzo su ogni strato:

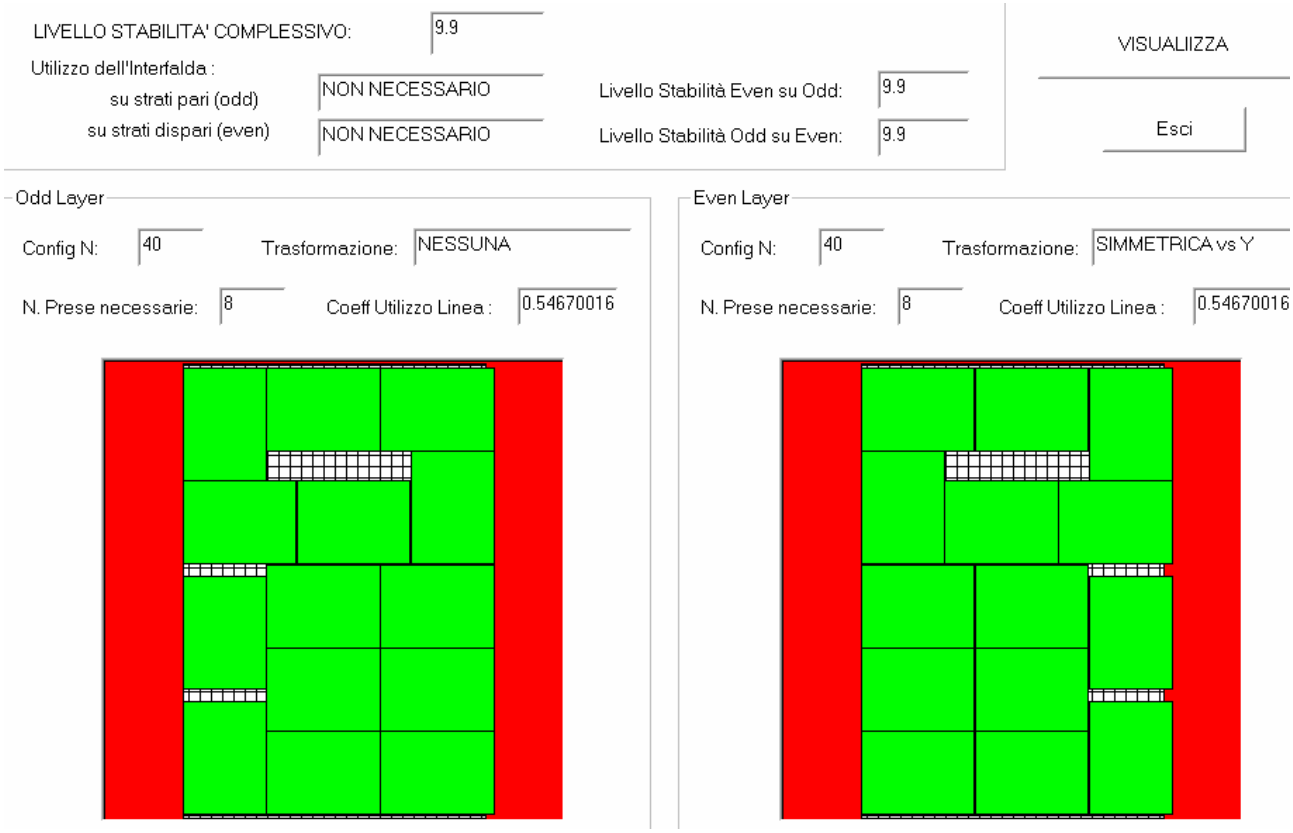


Figura 6.16: istanza [1200,800,300,220] con sforamento lungo l'asse 'x'

In questo modo si riesce ad inserire di 14 prodotti, anziché 13, grazie allo sforamento lungo 'x'. Si può notare che anche il livello di stabilità continua a rimanere molto buono.

Lo sforamento è un utile strumento per affinare il processo di pallettizzazione che non può assolutamente mancare in un programma ad esso relativo.

6.4. Analisi della stabilità

6.4.1. Stabilità nei prodotti 'rigidi'

Consideriamo l'istanza [1200,800,300,210]: un'impostazione fortemente orientata all'efficienza delle prese condurrà a introdurre un alto valore nello spazio consentito tra i pezzi per la funzione *perimetrizza*. Per la situazione illustrata nella figura seguente è stato impostato un valore pari a 80 mm. In questo modo il layout illustrato risulta essere una possibile configurazione.

Figura 6.17: istanza [1200,800,300,210]

Si osserva facilmente che una sovrapposizione con la sua trasformazione ruotata di 180° sarebbe molto buona dal punto di vista dell'intreccio dei pezzi, tuttavia il prodotto numero 5 si troverebbe sotto un bel buco, pari al 50% della propria superficie d'appoggio. In questo caso se non viene attivata l'opzione per i corpi rigidi, PALLETTIZZA non prenderà mai in considerazione questa soluzione dato che risulterebbe assolutamente instabile, in quanto verrebbe a mancare la condizione necessaria alla stabilità. Al contrario attivando tale opzione con uno spessore di contatto pari a 30 mm, il programma dopo aver verificato che il prodotto N.5, non ha effettivamente una superficie

d'appoggio sufficiente, passerà al controllo dell'appoggio sui quattro angoli recuperando in questo modo la condizione necessaria alla stabilità.

6.4.2. Superficie di contatto

Per la valutazione della formazione di eventuali colonne di pezzi il programma valuta che il pezzo sovrastante appoggi sul almeno due prodotti sottostanti per una certa percentuale di superficie. Di default tale superficie di contatto viene impostata pari al 15% della superficie totale d'appoggio del prodotto. Tuttavia tale impostazione in molti casi risulta essere una condizione piuttosto stringente che penalizza fortemente la valutazione della stabilità. Ad esempio l'istanza [220,160,50,30] non produrrebbe soluzioni con un buon valore di stabilità lasciando impostato il valore del 15% per la superficie di contatto. Portando questo parametro al 12% invece, PALLETTIZZA troverebbe come soluzione ottima quella illustrata in figura seguente:

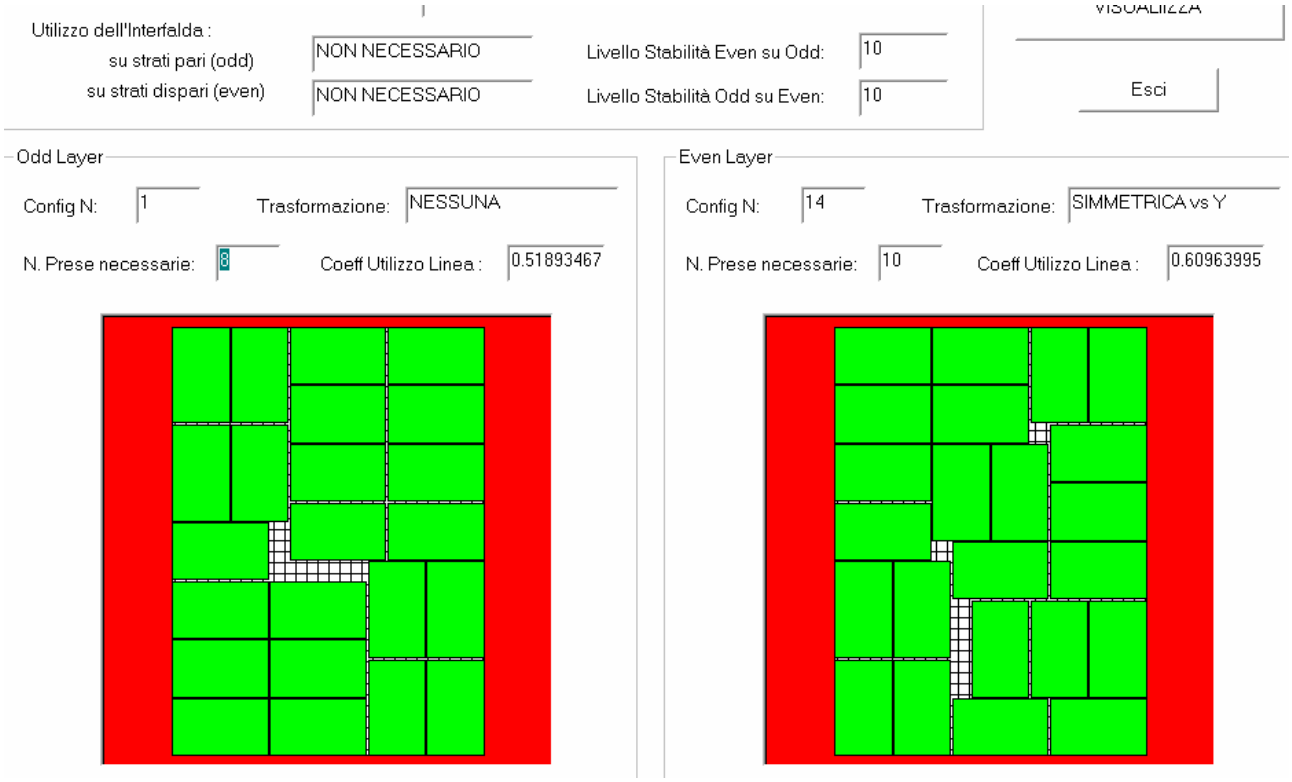


Figura 6.18: istanza [220,160,50,30]

Come si osserva la configurazione N.1 è realizzata con sole 8 prese, mentre la configurazione N.14 è realizzata con 10 prese. La soluzione che vedrebbe la coppia composta dalla configurazione N.1 e la sua ruotata di 180°, che risulterebbe molto favorevole dal punto di vista dell'efficienza delle

prese, viene scartata in quanto non soddisfa ancora il problema della formazione delle colonne di pezzi. Per ottenere questa soluzione, occorrerebbe impostare il parametro della superficie di contatto pari al 6%. Questa situazione si verifica in diverse istanze, ad esempio anche per [530,510,90,70] occorre abbassare il parametro di contatto al 10% per raggiungere dei buoni livelli di stabilità. In [23] si valuta di stabilire il parametro per la superficie di contatto in un valore compreso tra il 5% e il 15%, tuttavia per stabilire il valore di default più opportuno occorrerebbero degli ulteriori riscontri pratici.

6.5. Tempi computazionali

Quando le soluzioni generate sono molte, i tempi computazionali relativi al computo delle prese e alla generazione di coppie stabili, tendono ad allungarsi. In simili casi l'utente può cercare di sfoltire le soluzioni da analizzare tagliando quelle in cui, a priori, ritenga di non essere interessato a causa delle loro particolari caratteristiche.

6.5.1. Evitare i layouts con buchi di certe dimensioni

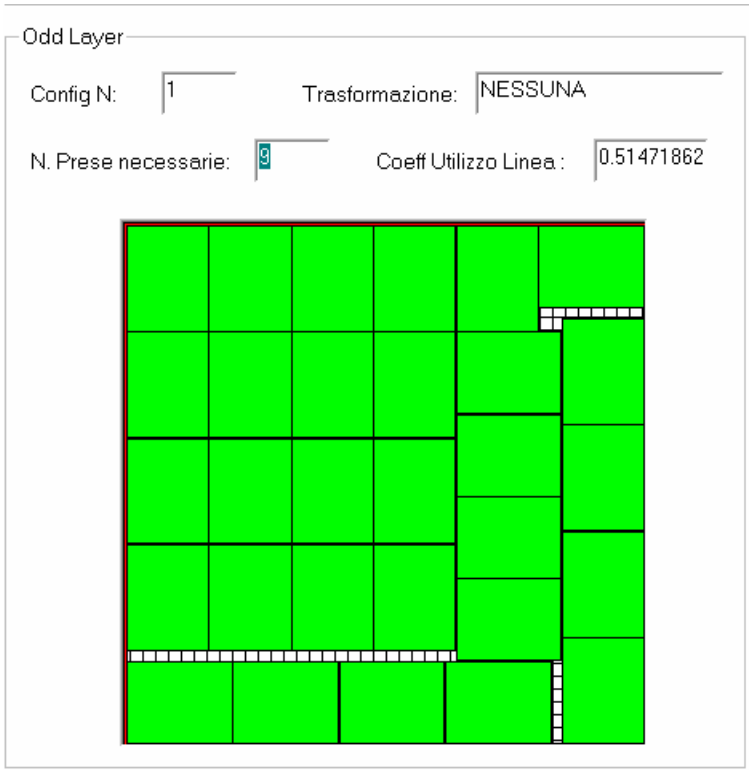


Figura 6.19: istanza [1000,1000,205,159]

Questa è una soluzione a prese minime prodotta da PALLETTIZZA per l'istanza [1000,1000,205,159]. Sopra i pezzi dell'ultima riga c'è un buco vuoto di dimensioni 636mmX21mm. Nel caso l'utente non voglia un simile buco tra le soluzioni, basterà che nella finestra per la generazione dei layers non venga selezionata l'opzione 'NON considerare holes' e si impostino i valori per i buchi dannosi di ampiezza pari 21mm e lunghezza 636mm. Pertanto sarà sufficiente spuntare l'opzione 'NON considerare layers con holes' e la soluzione qui sopra sarà automaticamente eliminata dall'analisi degli strati finali.

L'alternativa potrebbe essere quella di penalizzare i layouts con le caratteristiche indesiderate con penalità maggiori nelle valutazione della stabilità.

6.5.2. Tenere solo i layouts a prese minime

L'istanza [530,510,90,70] citata in [15], è un classico esempio in cui il vengono generate fin troppe soluzioni da analizzare: dall'algoritmo GeneraLayers vengono prodotti 170 layouts possibili. Lasciando tutte le impostazioni di default, di questi 170 layouts, ben 109 sono senza tagli a ghigliottina o buchi. Poichè questi sono già un numero considerevole, sarà inutile tenere anche i layouts che verranno sicuramente penalizzati nella valutazione della stabilità. Nel proseguo, per l'analisi delle prese, possiamo provare ad escludere tali layouts. Dall'analisi delle prese risulta che dei 109 layouts analizzati, 14 risultano essere ricostruibili con un numero minimo di prese pari a 13. Dal momento che anche in questo caso il numero di layouts a disposizione, risulta essere abbastanza elevato, si procede all'analisi della stabilità dei soli layouts a prese minime. In questo modo, impostando una superficie di contatto pari al 10%, PALLETTIZZA propone la seguente soluzione illustrata in Figura 6.20 per la pallettizzazione finale.

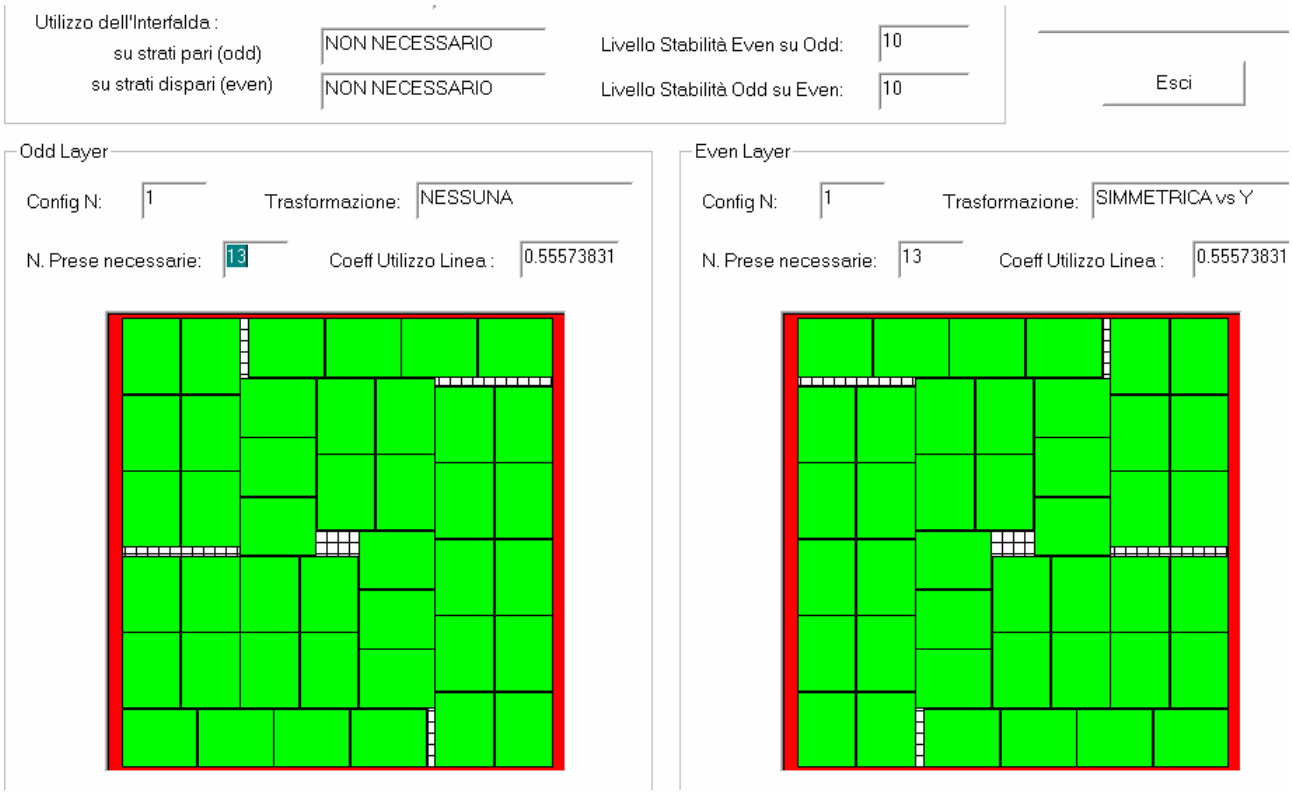


Figura 6.20: istanza [530,510,90,70]

Si osservi come i tagli effettuati abbiano prodotto ugualmente un risultato ottimo sia dal punto di vista della stabilità che da quello dell'efficienza delle prese con evidente risparmio di tempi computazionali.

6.6. Un compromesso tra il livello stabilità e l'efficienza delle prese

Come abbiamo già visto per la funzione perimetrizza, spesso una raffinatezza del livello di stabilità conduce ad un peggioramento dal punto di vista dell'efficienza delle prese. Consideriamo a riguardo l'istanza [1200, 800, 296, 196]: lasciando il 'livello di stabilità richiesto=10' la soluzione proposta da PALLETTIZZA è quella riportata in Figura 6.21:

Figura 6.21: istanza [1200,800,296,196]; soluzione con livello stabilità richiesto = 10

Come si vede la soluzione riportata ha effettivamente un livello di stabilità pari a 10. Tuttavia per la ricostruzione degli stati vengono impiegate 6 prese-rilasci per ciascuno, mentre i Layouts a prese minime impiegherebbero soltanto 5 prese-rilasci ciascuno. Effettivamente se abbassiamo il livello di stabilità richiesto al valore di 8, o anche 9, PALLETTIZZA produrrebbe come soluzione principale quella illustrata in Figura 6.22.

Figura 6.22: istanza [1200,800,296,196]; soluzione con livello stabilità richiesto = 8

Effettivamente questa soluzione era stata valutata anche nella precedente analisi, ma era stata pure scartata perché non raggiungeva i livelli di stabilità richiesti. In questo modo gli erano state preferite altre soluzioni. Ad esempio alla configurazione numero 2 non era stata neppure associata la sua versione ruotata, ma veniva preferito un altro gemello che più si avvicinava ai livelli di stabilità richiesti, vedi Figura 6.23. E' comunque evidente che il livello di stabilità ottenuto dalla configurazione numero 2 e la sua ruotata potrebbe essere ugualmente buono per molte applicazioni. Di conseguenza, poiché tale soluzione impiegherebbe meno prese-rilasci di tutte le altre, sarebbe da preferirsi.

In conclusione qualora l'utente non abbia bisogno di particolari parametri di stabilità, sarebbe opportuno tenere il livello di stabilità richiesto su valori di 8 o 9, altrimenti in alcuni casi si rischia di penalizzare troppo l'efficienza delle prese.

Figura 6.23: gemello associato alla configurazione n.2 con livello stabilità richiesto = 10

6.7. Tempi computazionali

Per stabilire l'efficienza computazionale dell'algoritmo sono stati valutati i tempi di elaborazione considerando diverse istanze al variare del numero di prodotti all'interno degli strati e al variare del numero di configurazioni da analizzare. I risultati sono riportati in
Tabella 1

Istanza	N. Prodotti per strato	N. Configurazioni analizzate	Tempo medio per la ricostruzione di un Layout (s)	Tempo medio per l'analisi della sequenza ottimizzata di prese-rilasci necessarie per ricostruire un singolo layout (s)	Tempo medio impiegato per determinare il gemello da associare ad un certo strato (s)
1200,800,300,220	13	8	0	0	0
1200,800,296,196	16	25	0	0	0
1200,800,300,150	20	353	0	0,2	0,8
220,160,50,30	23	18	0	0	0
1000,1000,400,100	24	166	0	0,2	0,7
1200,800,190,170	28	1	0	0	0
1000,1000,205,159	30	32	0	0,4	0,2
1200,800,180,160	32	35	0	0,4	0,1
1200,800,180,150	34	109	0	1,6	1,2
1200,800,170,150	37	1	0	2	0
1200,1200,260,140	38	105	0,1	0,2	0,5
570,440,120,50	41	18	0,2	0,3	0,1
530,510,90,70	42	109	0,2	0,6	1,1
520,330,90,40	47	19	0,4	1,2	0,3
560,520,120,50	48	19	1	1,8	0,2
430,260,70,30	53	1	3	6	0

Tabella 1: Tempi computazionali

I dati sono stati ottenuti utilizzando un Processore Intel Pentium IV a 2.66 GHz.

Visualizziamoli su dei grafici:

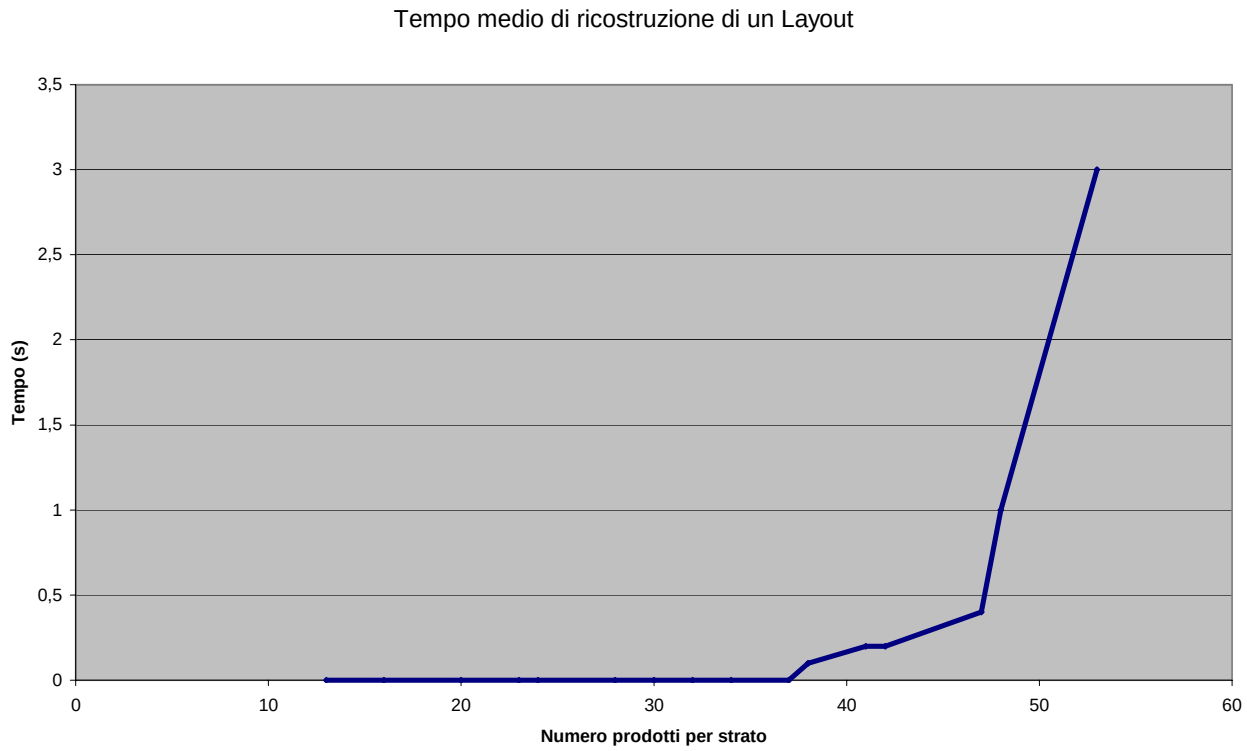


Grafico 1: Tempo medio di ricostruzione di un layout

Il Grafico 1 mostra l'efficienza dell'algoritmo per la ricostruzione dei layouts degli strati con numero massimo di prodotti Come è logico aspettarsi, i tempi computazionali salgono all'aumentare del numero di prodotti da inserire in quanto dovranno essere prese in considerazioni un maggior numero di casistiche.

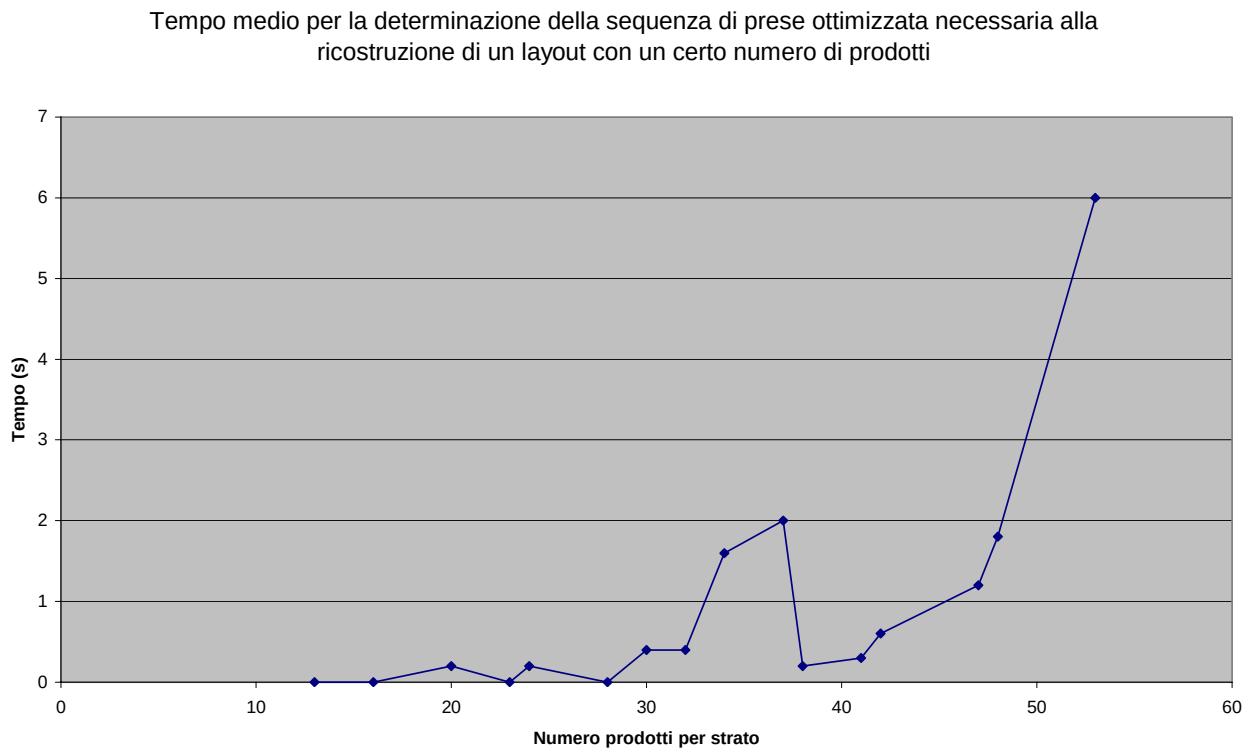


Grafico 2: Tempo medio per determinare la sequenza di prese-rilasci necessari alla ricostruzione di un layout

Dal Grafico 2 si osserva che il tempo impiegato per determinare la sequenza di prese-rilasci necessari alla ricostruzione fisica di un layout, tende ancora ad aumentare con il numero di prodotti presenti sullo strato, ma il suo andamento resta piuttosto altalenante e scarsamente prevedibile. Ciò è dovuto al fatto che la determinazione delle varie prese dipende anche dalla particolare configurazione del layout, e non soltanto dal numero di pezzi presenti in esso. E' chiaro che ad un aumento consistente del numero di prodotti per strato, corrisponderà pure un incremento delle prese-rilasci da dover effettuare, tuttavia la presenza di pochi prodotti in più o in meno, a seconda delle circostanze, potrebbe non produrre eccessivi cambiamenti dal punto di vista computazionale di tale analisi.

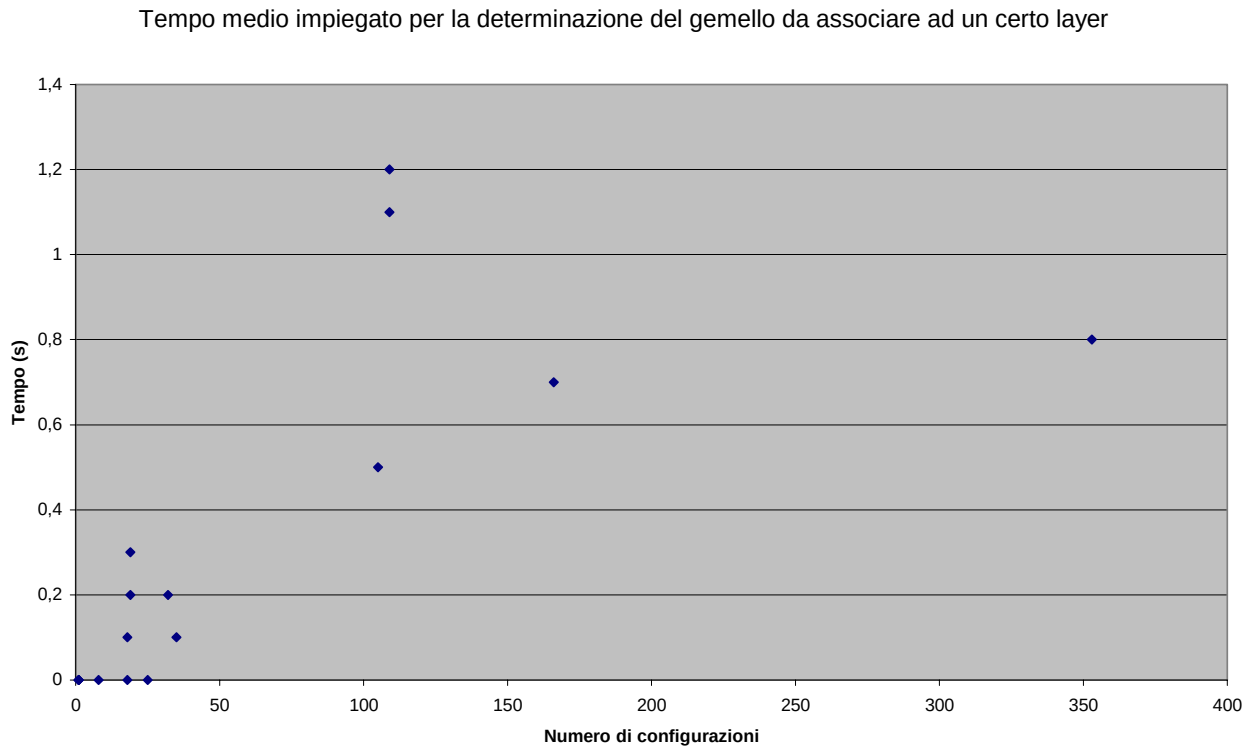


Grafico 3: tempi per l'analisi della stabilità

Per quanto riguarda la valutazione della stabilità invece, più che dipendere dal numero di prodotti inseriti, essa dipende dal numero di configurazioni che devono essere prese in considerazione. Si può intuire meglio questo risultato riordinando opportunamente i dati della Tabella 1 in base al numero di configurazioni analizzate:

Istanza	N. Configurazioni analizzate	N. Prodotti x strato	Tempo impiegato per l'analisi della stabilità (s)
1200,800,190,170	1	28	0
1200,800,170,150	1	37	0
430,260,70,30	1	53	0
1200,800,300,220	8	13	0
220,160,50,30	18	23	0
570,440,120,50	18	41	0,1
520,330,90,40	19	47	0,3
560,520,120,50	19	48	0,2
1200,800,296,196	25	16	0
1000,1000,205,159	32	30	0,2
1200,800,180,160	35	32	0,1
1200,1200,260,140	105	38	0,5
1200,800,180,150	109	34	1,2
530,510,90,70	109	42	1,1
1000,1000,400,100	166	24	0,7
1200,800,300,150	353	20	0,8

Tabella 2: istanze riordinate in base al numero di configurazioni prodotte

Come si può osservare dalla Tabella 2, finché il numero di tali configurazioni si mantiene basso, diciamo sotto le 30 configurazioni, indipendentemente dal numero di prodotti presenti per strato, i tempi computazionali sono estremamente veloci e molto simili tra loro.

Conclusioni

Con questo lavoro si è cercato di fornire una soluzione completa al problema della pallettizzazione valutando nel loro complesso i problemi di riempimento del volume, stabilità del pallet ed efficienza dell'utilizzo del sistema automatizzato. Il prodotto realizzato è stato finalizzato alle reali esigenze di un possibile utente cercando di renderlo sufficientemente veloce e facile da utilizzare. Le soluzioni prodotte soddisfano un numero sufficiente di casistiche, sia pratiche che teoriche, in cui vengono inseriti fino ad oltre 50 prodotti per strato. Il computo delle soluzioni avviene in tempi più che ragionevoli. Inoltre la possibile interazione con l'utente esterno garantisce al programma la possibilità di adattarsi alle situazioni più diverse in modo da riuscire a trovare sempre una soluzione accettabile per il problema proposto.

Possibili sviluppi futuri

Il programma è stato elaborato considerando tre aspetti fondamentali della pallettizzazione:

1. Generazione degli strati contenenti il numero massimo di prodotti
2. Conversione degli strati in funzione delle prese-rilasci per la loro ricostruzione sul pallet
3. Valutazione delle coppie stabili

Grazie alla programmazione orientata ad oggetti, le tre parti del programma possono essere rese indipendenti le une dall'altra. Ciò conferisce al software una maggiore versatilità che potrebbe facilitarne anche un suo eventuale ed ulteriore sviluppo futuro.

I. Prodotti a base quadrata

I prodotti a base quadrata sono un caso degenere dei più generici a base rettangolare. Per questa classe di problemi non ha senso utilizzare gli stessi algoritmi. Volendo pallettizzare con i prodotti esclusivamente ortogonali al pallet, l'unica soluzione è quella di disporre i pezzi l'uno accanto all'altro. Tutti gli strati risulteranno uguali, saranno tutti suddivisi in tagli a ghigliottina e quindi sarà sempre necessario l'uso dell'interfalda tra uno strato e l'altro. L'unico algoritmo ancora applicabile è quello relativo al computo delle prese, che si troverà a dover analizzare un unico grande blocco omogeneo.

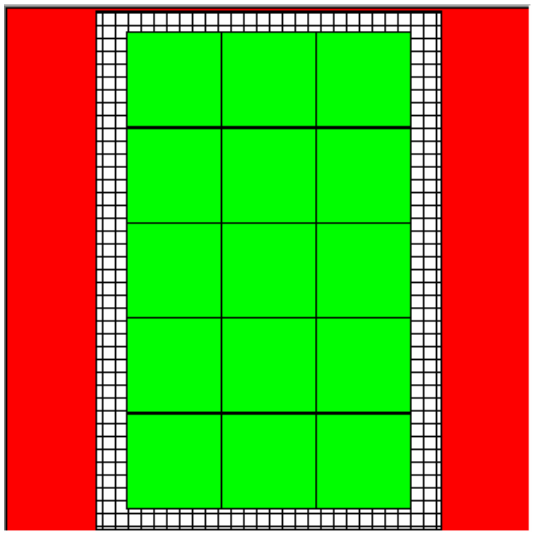


Figura 1: box quadrati, istanza [1200,800,220,220]

Se si vuole cercare di ottimizzare ancora di più lo spazio a disposizione, l'unica possibilità è quella di rinunciare all'ortogonalità dei pezzi. Gli studi di Erdos and Graham [30] e di Erich Friedman [31], ripresi poi da Michael J Kearney e Peter Shiu [32], hanno dimostrato come pallettizzando un prodotto a base quadrata, su un pancale, anch'esso a base quadrata, sia possibile migliorare il coefficiente di riempimento inserendo dei prodotti non ortogonali nella parte centrale del pallet, o lungo una diagonale.

Le figure che seguono mostrano alcuni esempi applicativi.

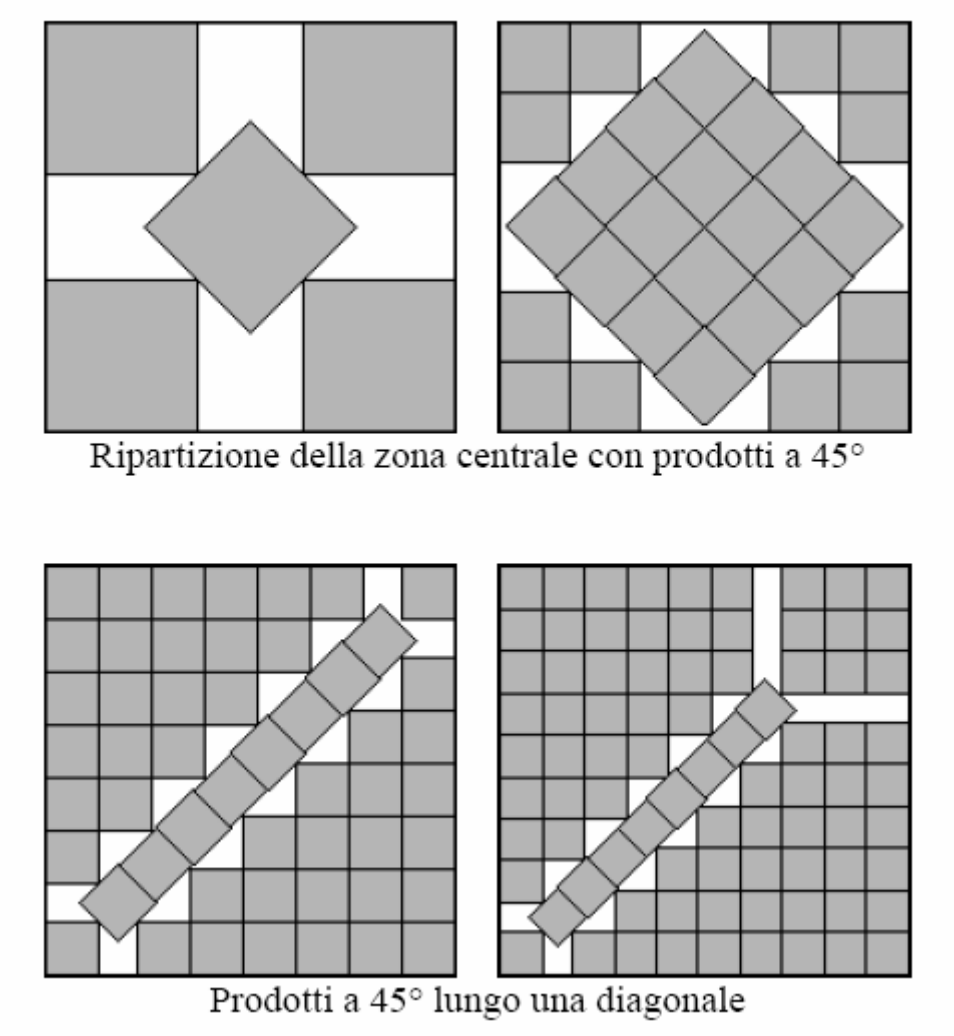


Figura 2: box quadrati con collocazione non-ortogonale

In molti casi si riesce veramente ad ottenere un'area persa sorprendentemente piccola. Tanto per capire, dividendo la lunghezza del lato del pancale per quella del prodotto, possiamo sempre ricondurci al caso di prodotti a base quadrata unitaria. In questo modo si ottiene

$$L = n + \sigma \qquad (1) \qquad \text{con } 0 \leq \sigma < 1 \text{ e } n \text{ intero naturale}$$

Con il banale metodo adottato da PALLETTIZA si inseriranno *n* prodotti con un'area persa

$$(n+\sigma)^2 - n^2 > 2n\sigma \qquad (2)$$

Invece adottando un riempimento dello strato non ortogonale si riesce a ottenere anche un'area persa di

$$O\left(n^{\frac{7}{11}}\right) \qquad (3) \qquad \text{se } n \text{ è grande}$$

Si osservi che tale miglioramento è limitato dalla necessità del pancale anch'esso di forma quadrata. Inoltre, dal punto di vista della stabilità non si aggiunge nulla di nuovo: anche ruotando gli eventuali strati si formerebbero delle colonne di prodotti e rimarrebbe quindi necessario l'uso dell'interfalda. L'idea potrebbe comunque essere la base di partenza per ulteriori nuovi sviluppi degli algoritmi di pallettizzazione.

II. Sistemi Pluri-Prodotto

In molti problemi di cutting&packing, si ha la necessità di inserire all'interno di un container prodotti, sempre a forma di parallelepipedo, ma magari con dimensioni diverse l'uno dall'altro. In letteratura si trovano già numerose ricerche sullo stato dell'arte. Le idee da sfruttare potrebbero essere analoghe a quelle considerate per la ricostruzione dei layouts mediante l'algoritmo esatto. Infatti si potrebbe cercare di valutare in maniera più rigorosa l'area persa ad ogni successivo inserimento, in modo da avere una stima sulla bontà o meno dell'inserimento del prodotto in questione [16]. Avendo a che fare con prodotti anche di altezza diversa, i problemi si complicherebbero lungo tutte e tre le dimensioni e quindi, piuttosto che l'analisi dell'area persa, si dovrebbe valutare il volume perso [27]. L'alternativa, forse più facile da implementare, potrebbe essere quella di suddividere lo spazio a disposizione in vari sottovolumi da riempire con pezzi di forma uguale [28] [29], vedi figura seguente.

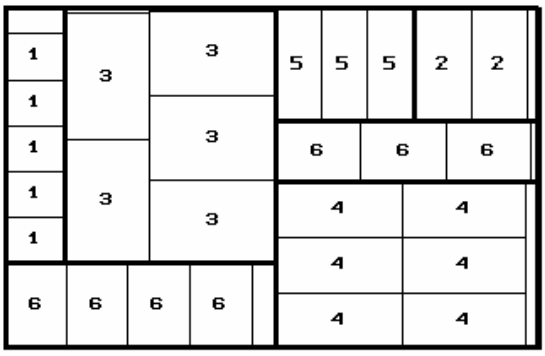


Figura 3: sistemi pluri-prodotto

In questo modo ciascun sottovolume potrebbe essere letteralmente 'pallettizzato' mediante l'ausilio del programma già implementato. L'unico problema si ridurrebbe alla valutazione di come dimensionare i vari sottovolumi, per ottenere un riempimento ottimizzato di tutto il sistema.

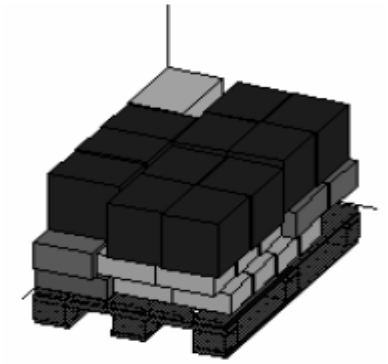


Figura 4: pallet completo multi-prodotto

III. Maggiori interazioni grafiche

Per rendere il prodotto sempre più dinamico e più appetibile dal punto di vista commerciale, potrebbe essere necessario fornire all'utente esterno una migliore gestione delle soluzioni finali. In tale ottica andrebbe valutato l'inserimento di ulteriori interazioni grafiche all'interno del programma. Ciò potrebbe essere fatto direttamente dalla finestra che visualizza i risultati ottenuti. Ad esempio l'utente potrebbe voler spostare determinati prodotti di uno strato, eventualmente anche ruotarli con inclinazioni non-ortogonali, e poi rivalutare i parametri del layout così modificato. L'implementazione di un'interfaccia grafica che consenta di compiere queste operazioni, magari

semplicemente selezionando il pezzo in questione col mouse, potrebbe costituire un decisivo miglioramento.

Bibliografia:

[1] Harald Dyckhoff, *A typology of cutting and packing problems* (1990), Eur. J. Opl. Res., Vol 44, pp. 145-159

[2] Christofides and Whitlock, *An algorithm for two-dimensional cutting problems* (1977), Operation Res., Vol. 13, pp. 33-40

[3] S.Barnet and G.J.Kynch, *Exact solution of a simple cutting problem* (1967), Opl Res. 15, 1051-1056

[4] Harold J. Stuedel, *Generating pallet loading patterns: a special case of the two dimensional cutting stock problem* (1979), Management Science, Vol. 25, pp. 997-1004

[5] Adrian Smith and Philippe De Cani, *An algorithm to optimize the layout of boxes in pallets* (1980), J. Opl Res. Soc., Vol. 31, pp 573-578

[6] E.Bischoff and W.B.Dowsland, *An application of the micro to product design and distribution* (1982), J. Opl Res. Soc., Vol. 33, pp 271-280

[7] J.E. Beasley, *An Exact Two-dimensional non guillotine cutting tree search procedure* (1983), Imperial college of science and technology, London, pp 49-65

[8] Kathryn A.Dowsland, *The three-dimensional pallet chart: an analysis of the factors affecting the set of feasible layouts for a class of two-dimensional packing problems* (1984), J. Opl Res. Soc., Vol. 35, N.10, pp 895-905

[9] Guntram Schetthauer and Johannes Terno, *The G4-Heuristic for the pallet loading problem* (1996), J. Opl Res. Soc., Vol. 47, pp 511-522

[10] R.Morabito and S.Morales, *A simple and effective recursive procedure for the manufacturer's pallet loading problem* (1998), J. Opl Res. Soc., Vol. 49, pp 819-828

[11] Young Gun and Maing Kyu Kang, *A fast algorithm for two dimensional pallet loading problems of large size (2001)*, Eur. J. Opl. Res., Vol 134, pp. 193-202

[12] Kathryn A.Dowsland, *An exact algorithm for the pallet loading problem* (1987), Eur. J. Opl. Res., Vol 31, pp. 78-84

[13] Wilson R.J., *Introduction to graph theory*, Longman, London, 1979

[14] Kathryn A.Dowsland, *A combinated data-base and algorithmic approach to the pallet loading problem* (1987), J. Opl. Res. Soc., Vol 38, N.4, pp. 341-345

[15] Subir Bhattcharya and Rahul Roy, *An exact depth-frist algorithm for the pallet loading problem* (1996), Eur. J. Opl. Res., Vol 110, pp. 610-625

[16] R.Yaman, B.Yargic,G.Celik, *Smart locating algorithm for cutting and packing problems*, Department of Industrial Engineering, Department of Mechanical Engineering, University of Balikesir Turkey.

[17] F.W. Barnes, *Packing the maximum number of m x n titles in a large p x q rectangle* (1978), Discrete Mathematics, Vol 26, pp. 93-100

[18] Josef Nelißen, *How to use structural constraints to compute an upper bound for the Pallet Loading Problem* (1995), Eur. J. Opl. Res., Vol 84, pp. 662-680

[19] Adam N.Letchford, Andrè Amaral, *Analysis of upper bounds for the Pallet Loading Problem* (2001), Eur. J. Opl. Res., Vol 132, pp. 582-593

[20] Kathryn A.Dowsland, *Efficient automated pallet loading* (1990), Eur. J. Opl. Res., Vol 44, pp. 232-238

[21] Boon J.Yuen and Ken V.Richardson, *Estabilishing the Optimality of sequencing heuristics for cutting stock problems* (1995) , Eur. J. Opl. Res., Vol 84, Iussue 3, pp. 590-598

[22] Sunderesh S.Heragu and Bulent Mazacioglu, *Solving order picking problem using simulated annealing based algorithms* (1992)

[23] H.Carpenter and W.B.Dowsland, *Pratical consideration of the pallet loading problem* (1985), J. Opl. Res. Soc., Vol 36, No.6, pp. 489-497

[24] W.B.Dowsland, *Improving palletisation efficiency – the theoretical basis and practical application* (1995), Int. J. Prod. Res., Vol 33, No. 8, pp 2213-2222

[25] Fuh-Hwa F Liu and C-J Hsiao, *A three-dimensional pallet loading method for single-size boxes* (1997), J. Opl. Res. Soc., Vol 48, pp. 726-735

[26] Jussy Jaakkola, *Timo Leipala, Olli Nevalainen, On the stability of Pallet loading problem* (2000), University of Turku, Departement of Mathematical Sciences and TUCS

[27] E.E.Bischoff, *Loading pallets with non-identical items* (1995), Eur. J. Opl. Res., Vol 84, pp. 681-692

[28] Guntram Schetthauer, Johannes Terno, Jan Riehme, *The solution of two stage guillotine cutting stock problems having extremely varying order demands* (1995), Eur. J. Opl. Res., Vol 91, pp. 543-552

[29] Guntram Schetthauer and Uta Sommerweib, *4-Block heuristic for the rectangle packing problem* (1998), Eur. J. Opl. Res., Vol 108, pp. 509-526

[30] P.Erdos and R.L. Graham, *On packing squares with equal squares (1975)*, J. Combin. Th. Ser. A **19**, pp. 119-123

[31] Erich Friedman, *Packing Unit Squares in Squares: A Survey and New Results* (2000) Stetson University, The Electronic Journal of Combinatorics 7.

[32] Michael J Kearney and Peter Shiu, *Effcient packing of unit squares in a square* (2002), Loughborough University.