TpaNestingOEM

TpaNestingOEM Library

Manuale d'uso

V. 4.1.0

16/03/2026

Tecnologie e Prodotti per l'Automazione

Indice

1 Presentazione

TpaNestingOEM.dll (alias TPA_N) è un prodotto realizzato per gli sviluppatori di software e può essere integrato come libreria per una architettura Windows a 32 e 64-bits.
TPA_N è una libreria per il nesting automatico di forme 2D complesse e permette lo sviluppo di applicazioni di ottimizzazione di piazzamenti 2D o di taglio di forme in vari settori applicativi.

La libreria deve essere integrata come componente del vostro prodotto.

Un applicativo dimostrativo della libreria è disponibile al sito www.tpaspa.com/oem-nesting-library.
Occorre però precisare che questa applicazione non espone tutte le funzionalità disponibili in TPA_N.

1.1 Versioni

Versione	*Data di rilascio*	*Commenti*
1.0.0	14-04-2021	
1.5.0	11-01-2022	Metodo *Compute*: modificato prototipo Aggiunta proprietà: *Version* Aggiunto metodo: *SaveSolutionDXF*
1.8.0	14-06-2022	Aggiunta proprietà: *TimeCompute*, *TimerProgres*
1.9.0	29-09-2022	Modifiche di manutenzione del prodotto
3.0.0	20-11-2023	Revisione complessiva del prodotto
3.2.0	17-06-2024	Struttura *NestBox*: aggiunti campi relativi agli ingombri del piazzamento Aggiunta gestione di proprietà: *UseOnlyExtraPart* Sono stati migliorati i criteri di applicazione del distanziamento dei piazzamenti dai bordi dei fogli Sono stati migliorati i criteri di calcolo del rendimento di ottimizzazione *Square* di parti non rettangolari Sono stati risolti problemi segnalati o emersi in fase di verifica interna del prodotto
3.3.0	11-06-2024	Aumentato il numero di piazzamenti che è possibile richiedere per l'utilizzo di una parte o cluster (vedi costanti: MAX_ROW_N, MAX_CLUSTER_N) Modifiche di manutenzione del prodotto
3.5.0	14-04-2025	Modifiche di manutenzione del prodotto
3.6.0	20-10-2025	Modifiche di manutenzione del prodotto
4.0.0	20-10-2025	Modifiche di manutenzione del prodotto Nuova gestione del sistema delle licenze
3.7.0	16-03-2026	Aggiornamento della versione (3.6.0) Aggiunta funzionalità step-by-step anche per ottimizzazione Square Aggiunta funzionalità di raggruppamento di parti Aggiunta funzionalità di nesting incrementale Struttura *NestSheet*: aggiunti campi Struttura *NestBox*: aggiunti campi Ottimizzazione Square: aggiunta gestione delle isole di lastre Eliminate proprietà obsolete
4.1.0	16-03-2026	Aggiornamento della versione (4.0.0) Le specifiche della libreria sono allineate alla versione (3.7.0)

1.2 Licenza

A partire dalla versione 4.0.0, l'operatività di TPA_N richiede la verifica di licenza software.
Nel presente documento la dicitura *chiave* è utilizzata come sinonimo di *licenza software*.
La gestione della licenza software è svolta da un applicativo dedicato, a cui si rimanda per informazioni dettagliate.

La chiave può assegnare abilitazioni dell'ottimizzazione a due livelli: *Square* (codice d'ordine: SW2.4.01) e *True Shape* (codice d'ordine: SW2.4.02).
L'abilitazione per ottimizzare *True Shape* comprende quella *Square* e rende disponibile la funzionalità accessoria di importazione di geometrie da file in formato DXF.

L'importazione di file in formato DXF interpreta solo la versione ASCII.

Se non altrimenti specificato, di seguito si intende operare con abilitazioni complete.

In assenza di chiave non sarà possibile eseguire alcuna ottimizzazione di progetto.

1.3 Requisiti di sistema

I requisiti minimi del PC su cui si andrà a installare l'applicativo sono i seguenti:

- Sistema operativo: 64bit Windows 10 o superiore
- Processore: Dual core (raccomandato Quad core): più veloce è la CPU, più veloce sarà il calcolo dell'ottimizzazione
- Memoria: 1 GB o più
- Estensioni di Windows necessarie: .NET framework 4.8 o superiore
- Accesso come amministratore è richiesto per l'installazione della suite.

1.4 Proprietà e diritti d’autore

1.5 Contatti

Per ulteriori informazioni, contattate:
T.P.A. Srl Tecnologie e Prodotti per l'Automazione - Via Carducci, 221 - 20099 Sesto S. Giovanni (MI) - ITALIA
Tel. +390236527550
e-mail: info@tpaspa.it

oppure visitate il nostro sito web:
www.tpaspa.com

2 Guida utente

TPA_N assegna la classe ***Nesting*** nel namespace ***TpaNestingOEM***.

TPA_N realizza il calcolo dei piazzamenti di *forme* in aree di piazzamento rettangolari o irregolari.
Una *forma* è una geometria poligonale:

- la forma più semplice è definita come rettangolo
- una forma può definire una geometria esterna e una o più geometrie interne (isole).

Una *parte* è una singola entità che può essere realizzata con piazzamento su un'area.
Le aree di piazzamento sono chiamate *fogli*.

Una *parte* ha una geometria, generalmente assegnata come *forma*. La forma assegnata ad una *parte* ha un solo percorso esterno e può avere isole: aree di sfrido interne, eventualmente utilizzabili per il piazzamento di altre *parti* di dimensione inferiore.
Un *foglio* assegna una lastra di materiale dove le parti sono piazzate. Anche i fogli hanno una geometria, generalmente assegnata come *forma*: generalmente i *fogli* sono semplici rettangoli, ma è possibile assegnare forme irregolari ed eventuali aree di vincolo interne, non utilizzabili per il piazzamento di parti.

L'assegnazione delle *parti* permette di definire trasformate della geometria di base, come rotazione e/o specularità, oltre che assegnazioni di carattere generico (quantità disponibile, priorità di piazzamento) o generalmente tecnologico (spessore, diametro dell'utensile di taglio, materiale, colore, venatura, …). Per ogni tipologia di parte si definiscono la quantità richiesta da piazzare ed una eventuale quantità massima piazzabile, a riempimento dei fogli già utilizzati.

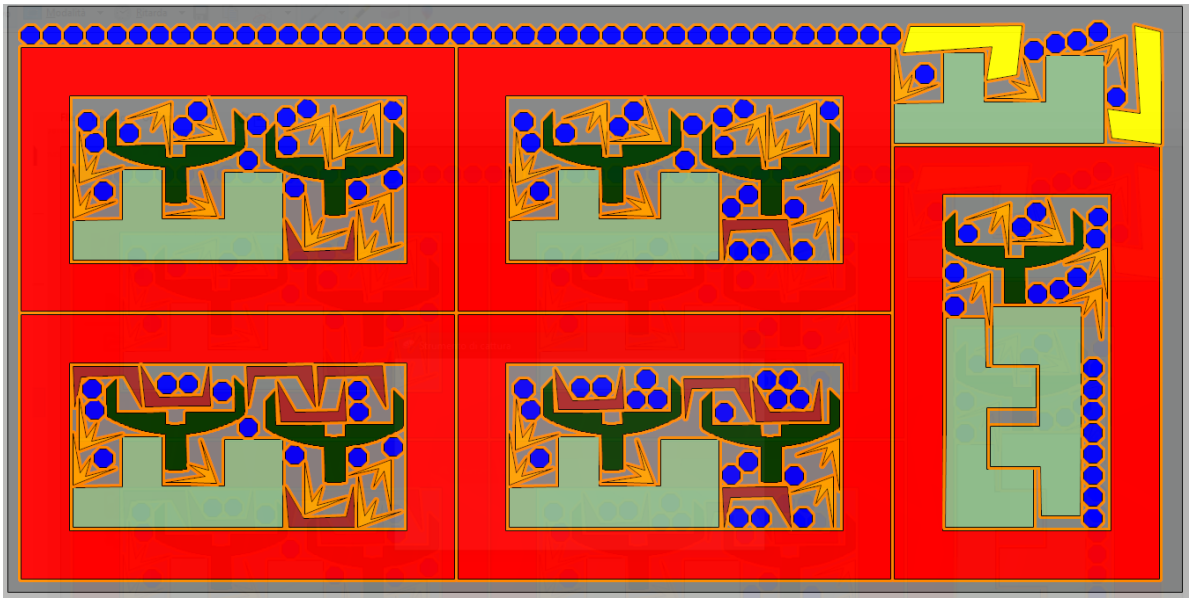
È possibile creare raggruppamenti di due o più parti e utilizzare direttamente i gruppi per il piazzamento: parliamo di *abbinamenti* o *cluster manuali*.

L'assegnazione dei *fogli* permette di definire impostazioni di carattere generalmente tecnologico (spessore, materiale, colore, venatura, …). Per ogni tipologia di foglio si definisce la quantità disponibile.

Le informazioni di carattere generalmente tecnologico permettono di applicare alla procedura di nesting condizioni di corrispondenza e/o di filtri tra *parti* e *fogli*.

Il *risultato* dell'ottimizzazione di nesting assegna la posizione e rotazione di tutte le *parti* che sono piazzate sui *fogli disponibili*. Un *risultato* del nesting può assegnare uno o più *fogli*: ogni *foglio* contiene il piazzamento di almeno una *parte*.

La figura illustra un foglio rettangolare ottenuto con ottimizzazione *True Shape*.



2.1 Tipologie di ottimizzazione del Nesting

L'insieme delle procedure di calcolo che porta a una soluzione di nesting è indicata come *fase di ottimizzazione*. TPA_N gestisce due modalità di ottimizzazione: *Square* e *True Shape*.

Ottimizzazione Square

- Gestisce il piazzamento di *parti* rettangolari in *fogli* rettangolari
- il piazzamento di una parte può applicare rotazioni di 0° o 90° (in senso antiorario) e trasformate speculari
- è possibile calcolare ottimizzazioni successive, valutabili come alternative equivalenti, oppure richiedere direttamente il migliore risultato ottenibile.

Se tutte le *parti* e i *fogli* sono assegnati come semplici rettangoli, la libreria assume ottimizzazione *Square* come default.

In caso di forzatura di ottimizzazione *Square*:

- le *parti* assegnate con un profilo geometrico sono considerate per il rettangolo di ingombro del profilo stesso e i profili assegnati come isole sono ignorati
- un *abbinamento manuale* è considerato per il rettangolo di ingombro complessivo delle parti che lo compongono
- i *fogli* assegnati con un profilo geometrico sono considerati per il rettangolo di ingombro del profilo stesso: in questo caso, la soluzione di ottimizzazione può eseguire dei piazzamenti in aree esterne al foglio assegnato
- anche i profili assegnati come aree di vincolo di un *foglio* sono considerati per il rettangolo di ingombro dei profili stessi: in questo caso, le aree di vincolo sono sovrastimate.

Ottimizzazione True Shape

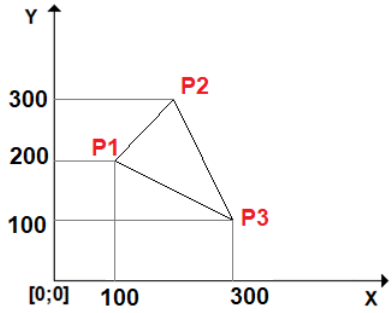
- Gestisce il piazzamento di *parti* e *fogli* comunque assegnate
- una *parte* può contenere una o più isole (aree di sfrido interne), eventualmente utilizzabili per il piazzamento di altre *parti*
- un *foglio* può contenere aree di vincolo, nelle quali è escluso il piazzamento di parti
- il piazzamento di una *parte* può applicare rotazioni variabili ed eventuali trasformate speculari
- il piazzamento delle *parti* su un *foglio* rispetta le forme di parti e foglio, con il risultato di ottimizzare l'utilizzo del materiale
- è possibile calcolare ottimizzazioni successive, valutabili come alternative equivalenti.

2.2 Unità e coordinate

In TPA_N i valori metrici (coordinate, dimensioni) sono espressi nell'unità definita come [mm] o [inch] nella chiamata a funzione *IniSettings(...)*.

TPA_N opera in un sistema 2D di coordinate cartesiane:
- il punto di coordinate [0;0] è posizionato sull'angolo inferiore sinistro
- X è l'asse orizzontale, positivo verso destra
- Y è l'asse verticale, positivo verso l'alto.

In figura un semplice esempio di poligono assegnato su 3 punti: P1(100;200), P2(200;300), P3(300;100)



Verrà fatto riferimento a caratteristiche geometriche notevoli di un poligono. Vediamo come sono definite:
- *rettangolo di ingombro*: area rettangolare che contiene completamente il profilo sul piano XY
- *vertice LB*: spigolo (inferiore sinistro) del rettangolo di ingombro:
  - indichiamo il vertice opposto a LB come RT (superiore destro)
  - i termini (sinistro, destro) sono riferiti all'asse orizzontale (asse X)
  - i termini (inferiore, superiore) sono riferiti all'asse verticale (asse Y)
  - come è evidenziato dalla figura: i punti estremi del rettangolo di ingombro non appartengono necessariamente al profilo.

Come in figura:
- il rettangolo di ingombro è individuato con i due punti estremi: LB={100; 100}, RT={300; 300}.

Tutti gli angoli sono espressi in unità di grado e decimo di grado. Dove è significativa l'interpretazione della rotazione:
- a un angolo positivo è associata rotazione antioraria
- a un angolo negativo è associata rotazione oraria.

2.3 Dimensioni del Nesting

TPA_N può operare con due differenti tipi di fogli, in base al settaggio *OneNestingDimension*
- *(false)* 2D: i fogli assegnano una parte di piano su cui il nesting è bidimensionale
- *(true)* 1D: il nesting è su supporto lineare e i fogli sono barre, profilati, tubi.

Nesting 2D

L'ottimizzazione opera sullo spazio 2D come delimitato dai fogli. La selezione corrisponde al funzionamento di default (*OneNestingDimension=false*).
Applicazioni tipiche riguardano il taglio di lastre di forme regolari o irregolari dei materiali più svariati: metallo, plexiglass, legno, pelle, carta, tessuto.

Nesting 1D

Applicazioni tipiche riguardano il taglio di:
- barre, tubi o profilati di metallo
- travi di legno.

La selezione forza l'utilizzo di *Direzione verticale* ma non limita né piazzamenti consecutivi in progressione orizzontale né l'altezza dei fogli all'altezza delle parti.
In figura un esempio di soluzione di ottimizzazione di una trave: i due triangoli riportati sulla parte sinistra evidenziano come la procedura di ottimizzazione mantenga comunque attive le logiche generiche di piazzamento in piano bidimensionale.

La selezione indirizza TPA_N nella scelta di criteri che sono studiati per ottenere risultati migliori in caso di piazzamenti su supporto lineare. Una selezione inappropriata può determinare risultati non ottimali, specie dal punto di vista del tempo di calcolo.
Impostazioni da associare alla selezione possono riguardare:
- selezione della tipologia dell'utensile di taglio delle parti (settaggio generale: *OneCutterDimension*)

- impostazioni relative al sormonto dei profili di taglio ai bordi della lastra (settaggi generali: OverrunSheet, MaximizeOverrunSheet)
- criterio di applicazione speculare delle parti (settaggio generale: ModeMirrorPart)

L'esempio sopra riportato:
- può lavorare con un utensile di taglio di tipo lineare (es.: lama)
- i tagli delle parti escludono i rifili orizzontali, lungo la parte superiore e inferiore della trave: i tagli possono quindi essere applicati del tutto esterni ai bordi della trave
- la trave non ha un lato buono (la parte superiore e inferiore si equivalgono): una parte può quindi essere applicata in modalità normale o speculata, in base all'efficienza maggiore.

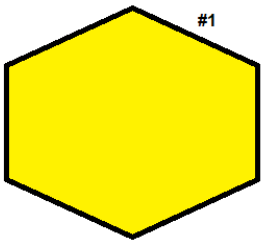
2.4 Definizione di profili

Una *parte* o un *foglio* è assegnato tramite un profilo geometrico primario ed eventuali profili secondari (isole per una parte o aree di scarto per un foglio): il profilo geometrico più semplice è un rettangolo, assegnabile direttamente con le dimensioni di *lunghezza* e *altezza*.

Un profilo geometrico può essere composto da tratti lineari o curve (archi) ed è comunque utilizzato *chiuso*, con eventuale chiusura con un tratto lineare. Come da figura precedente: il profilo è assegnato con i tre punti (P1, P2, P3) ed è chiuso con il tratto lineare che unisce P3 a P1.

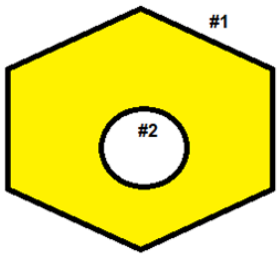
Ogni profilo assegna un'area *netta*:
- di taglio, per le parti
- di confine, per i fogli.

Se una parte (o foglio) assegna una geometria, le dimensioni primarie 2D della parte non sono utilizzate (campi *L* e *H* in struttura ***NestPart***/***NestSheet*** di assegnazione della parte/foglio).



La figura rappresenta il caso di una parte senza isole:

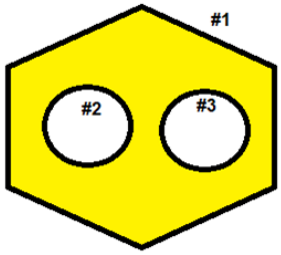
- #1 contrassegna il profilo primario esterno.



La figura rappresenta il caso di una parte con 1 isola:

- #1 contrassegna il profilo primario esterno
- #2 contrassegna un profilo di sfrido interno.

In assegnazione di parte: i profili di sfrido possono essere utilizzati per piazzamenti di altre parti.
In assegnazione di fogli: i profili di sfrido rappresentano aree non utilizzabili per piazzamenti.



La figura rappresenta il caso di una parte con due isole:

- #1 contrassegna il profilo primario esterno
- #2 e #3 contrassegnano i due profili di sfrido interni.

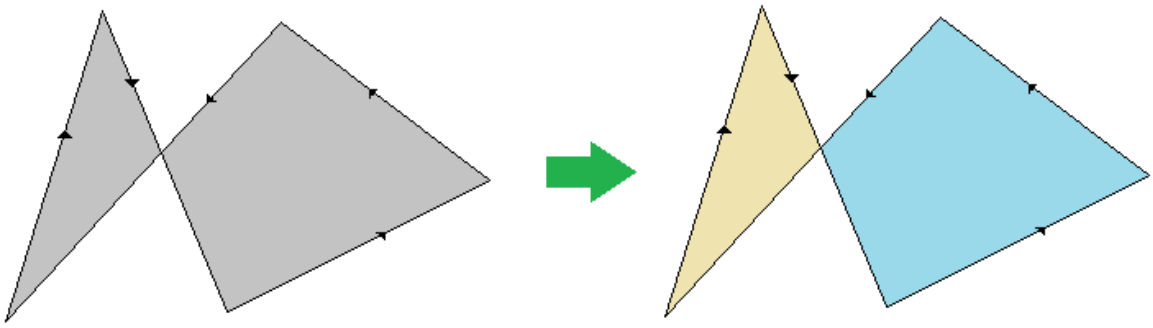
Il rettangolo di ingombro di una geometria può essere assegnato dovunque, in area X e/o Y positiva o negativa: assegnare una geometria in area XY positiva e ingombro minimo in (0;0) può essere una comodità di programmazione, ma non costituisce una condizione necessaria.

Semplificazione dei profili

Un profilo geometrico è valido se assegna una geometria *semplice*: senza tratti *nulli* o *intersecanti* e che chiude un'area non nulla.

Al fine di ottenere una geometria *semplice*, ogni profilo è elaborato da TPA_N con:
- rimozione di vertici coincidenti (a meno di un epsilon di risoluzione lineare) o allineati
- rimozione di situazioni di intersezioni, con eventuale scissione in più profili.

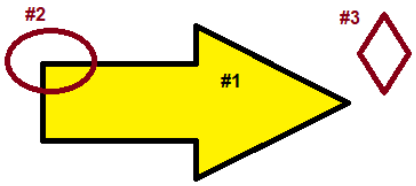
In figura un esempio di profilo con tratti intersecanti:
- a sinistra: il profilo originale
- a destra le due aree colorate evidenziano come il profilo sia suddiviso in due profili, ognuno dei quali deve poi definire una geometria semplice
- in assegnazione di
  - parte: i due profili semplici ottenuti sono opportunamente uniti e utilizzati come un unico elemento
  - foglio: saranno possibili piazzamenti separati entro ogni geometria semplificata



Profili non riconducibili a una geometria valida sono scartati dalla procedura di nesting, con possibile esclusione dal calcolo dei piazzamenti.

Attenzione va posta nella assegnazione di profili interni (isole delle parti o aree di vincolo dei fogli):
- **devono** essere completamente interne al corrispettivo profilo primario
- in caso di isole di una parte: sono utilizzabili per il piazzamento di altre parti solo se non contengono o si intersecano a loro volta con altre isole.

La figura illustra una situazione di assegnazione errata:



- #1 contrassegna il profilo primario (l'area interna è di colore: giallo)
- #2 contrassegna un profilo di sfrido che *interseca* con il profilo primario
- #3 contrassegna un profilo di sfrido che è *esterno* al profilo primario

Entrambi i profili di sfrido saranno ignorati dalla procedura di nesting. In particolare: l'area occupata dai due profili di sfrido non potrà essere in alcun modo rispettata nella valutazione del piazzamento della parte su un *foglio*.
Una situazione come quella qui illustrata comporta la sicura sovrapposizione di piazzamenti di altre parti nell'area dei profili indicati come (#2, #3): è responsabilità dell'applicazione chiamante verificare e/o segnalare situazioni simili.

Approssimazioni dei profili

I tratti curvi di un profilo sono approssimati in una serie di tratti lineari, con imprecisione massima tollerabile definita di 0.3 mm:
- ciò significa che la distanza massima che il percorso appiattito devierà dall'arco *reale* non sarà superiore a 0.3 mm
- il numero di tratti in cui un arco è approssimato varia in base al raggio dell'arco: aumenta al diminuire del raggio.

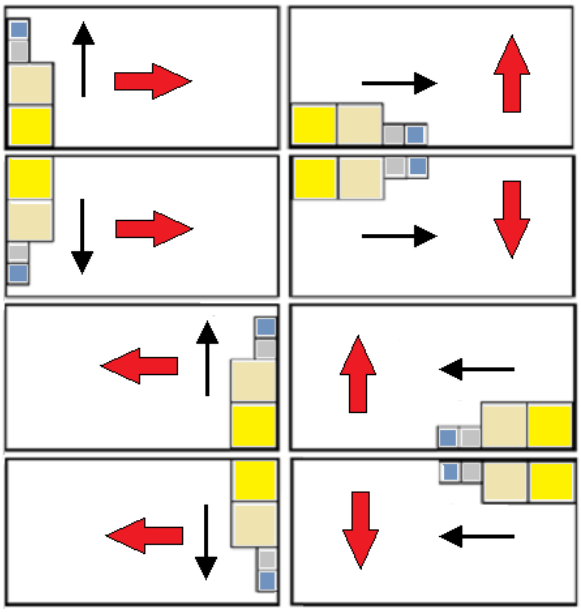
Il valore di tolleranza qui introdotto è indicato come *Errore cordale*.

L'approssimazione di uno o più tratti di un profilo comporta l'aggiunta di una distanza di sicurezza (almeno uguale all'*Errore cordale*) tra piazzamenti contigui o tra le parti e il bordo del foglio.

2.5 Direzione e Vertice di sviluppo del nesting

La Direzione del Nesting assegna la direzione di avanzamento per i piazzamenti, a riempimento dei fogli:
- *direzione orizzontale*: i piazzamenti sono effettuati per primi in direzione verticale (in figura: casi a sinistra, con freccia rossa orizzontale)
- *direzione verticale*: i piazzamenti sono effettuati per primi in direzione orizzontale (in figura: casi a destra, con freccia rossa verticale)

Il Vertice del Nesting assegna il vertice di partenza per i piazzamenti, tra i quattro valori possibili:



- 0 = Sinistro-Inferiore (in figura: casi sulla prima riga dall'alto)
- 1 = Sinistro-Superiore (in figura: casi sulla seconda riga dall'alto)
- 2 = Destro-Inferiore (in figura: casi sulla terza riga dall'alto)
- 3 = Destro-Superiore (in figura: casi sulla quarta e ultima riga dall'alto)

Le informazioni corrispondono alle assegnazioni generali di progetto Direction e Corner.

TPA_N utilizza sempre l'impostazione del Vertice come assegnata.
È invece possibile che TPA_N effettui tentativi di ottimizzazione con valore modificato di Direzione, al fine di ottenere soluzioni migliori.

2.6 Gruppi e filtri di corrispondenza

L'assegnazione delle *parti* e dei *fogli* prevede impostazioni di carattere generalmente tecnologico, allo scopo di applicare condizioni di corrispondenza e/o di filtri tra *parti* e *fogli*, anche se non ne costituisce la normale condizione di utilizzo.

Esaminiamo dapprima i settaggi che concorrono ad applicare condizioni di corrispondenza.

I settaggi interessati corrispondono a campi in strutture NestPart e NestSheet:

- *S*: (tipo: double) spessore
- *Fiber*: (tipo: intero) assegnazione generica di materiale
- *Colour*: (tipo: intero) assegnazione generica di colore
- *Grain*: (tipo: intero) direzione di venatura o grana.

Campo S: la corrispondenza è valutata sull'uguaglianza dei valori impostati, a meno di un epsilon di confronto lineare corrispondente al settaggio MinResolution.
Ulteriori settaggi abilitano l'applicazione all'utilizzo dei campi *Fiber*, *Colour* e *Grain* (MatchType, MatchColor e MatchGrain).

Esempi di assegnazione di gruppi di corrispondenza sullo spessore di parti e fogli:

- sono assegnate due parti (di identificativi: 1, 2) con campo *S*=18.0
- è assegnata una parte (di identificativo: 3) con campo *S*=25.0
- è assegnato un foglio (di identificativo: 1) con campo *S*=0.0
- è assegnato un foglio (di identificativo: 2) con campo *S*=18.0

la situazione determina l'assegnazione di tre gruppi di corrispondenza:

- gruppo 1, in corrispondenza a *S*=18.0
- gruppo 2, in corrispondenza a *S*=25.0
- gruppo 3, in corrispondenza a *S*=0.0.

Ogni gruppo è ottimizzato in modo separato: equivale a richiedere una ottimizzazione per tre differenti progetti. Dall'esempio sopra esposto, è evidente come il solo gruppo 1 possa determinare una soluzione di Nesting, permettendo l'associazione tra parti e fogli: le parti di identificativi (1, 2) potranno essere piazzate in fogli di identificativo 2.

In caso di assegnazione di valori multipli (ad esempio: anche per il materiale) le associazioni di corrispondenza tra parti e fogli possono aumentare di complessità. Un esempio:

- gruppo 1, in corrispondenza a *S* = 18.0 e *Fiber*=0
- gruppo 2, in corrispondenza a *S* = 18.0 e *Fiber*=1
- gruppo 3, in corrispondenza a *S* = 25.0 e *Fiber*=0.

Il controllo della Venatura

Il campo compare in strutture NestPart e NestSheet, in corrispondenza all'assegnazione della direzione di venatura o grana:
- l'assegnazione avviene in campo *Grain* delle strutture
- il significato tecnologico dell'informazione dipende dal materiale delle lastre (esempi: legno, impiallacciato, metallo).

I valori assegnabili sono tre:

- *0 (None)*: non assegna venatura
- *1 (X)*: venatura lungo la direzione orizzontale
- *2 (Y)*: venatura lungo la direzione verticale.

L'assegnazione di venatura non porta necessariamente alla determinazione di gruppi separati di corrispondenza: parti con stesso valore in campo *Grain* possono essere piazzate in lastre di gruppi differenti e/o con stesso o differente valore di campo *Grain*. Il comportamento è regolato dalla proprietà MatchGrain.

Se è *MatchGrain=true*: ogni valore di *Grain* determina una corrispondenza separata:
- una parte con *Grain=(1, 2)* può essere piazzata solo in fogli con stessa venatura
- una parte con *Grain=0* può essere piazzata solo in fogli senza venatura.

Vediamo quali criteri sono applicati in caso di *MatchGrain=false*:
- una parte con *Grain=(1, 2)* può essere piazzata con rotazione qualunque in foglio con *Grain=0*
- una parte con *Grain=(1, 2)* può essere piazzata in foglio con *Grain=(1,2)* con rotazione tale da rispettare la direzione assegnata per entrambi
- una parte con *Grain=0* può essere piazzata con rotazione qualunque indipendentemente dal campo *Grain* dei fogli.

Se una parte con *Grain=(1, 2)* può essere piazzata in lastre differenti per venatura:
- non viene garantito un piazzamento privilegiato in lastra con stessa venatura
- non viene applicato il settaggio *RctMinimize* attivo, se c'è la possibilità di utilizzo della parte in un tipo di foglio con venatura assegnata.

2.7 Trasformate applicate alle parti

Per le parti è possibile assegnare informazioni relative a due trasformate geometriche:
- speculare
- rotazione.

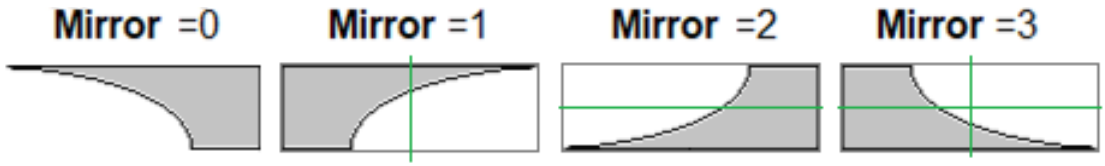
Speculare delle parti

TPA_N supporta due modi per l'applicazione speculare delle parti, in corrispondenza al settaggio *ModeMirrorPart* (di tipo ***boolean***).

Forzatura dello speculare

Il funzionamento di default corrisponde a *ModeMirrorPart=false*.
Per ogni parte è possibile richiedere il piazzamento speculato, lungo uno o entrambi gli assi coordinati, con selezione possibile tra quattro scelte:
- *0 (None)*: la parte non è speculata
- *1 (X)*: la parte è speculata attorno all'asse di mezzeria verticale del proprio rettangolo di ingombro
- *2 (Y)*: la parte è speculata attorno all'asse di mezzeria orizzontale del proprio rettangolo di ingombro
- *3 (X+Y)*: somma le due precedenti opzioni.

L'assegnazione per le parti corrisponde al campo *Mirror* in struttura NestPart:



La simmetria è applicata prima di una eventuale rotazione della parte.
La modalità corrisponde al caso di lastra con un *lato* buono di taglio: tipicamente il lato superiore.
Applicazioni specifiche utilizzano supporti di tipo non lineare: lastre di legno o metallo, pelle, tessuto.

Speculare controllato

Il funzionamento corrisponde a *ModeMirrorPart=true*.
Lo speculare può essere applicato a una parte se non è possibile il piazzamento in modalità normale, anche provando rotazione differenti.
La selezione è su quattro scelte, anche se ai fini dell'utilizzo può essere sufficiente la scelta tra i primi due valori:
- 0 (None): non è possibile speculare la parte
- 1 (X): è possibile speculare la parte.

Se risulta selezionata la dimensione del nesting lineare (*OneNestingDimension=true*): i tentativi di piazzamento speculare della parte sono eseguiti comunque, anche se una parte risulta piazzabile in modalità normale.
La modalità corrisponde al caso di lastra senza un *lato* buono di taglio: il lato superiore e inferiore si equivalgono. Applicazioni specifiche possono utilizzare supporti di tipo lineare: barre, travi.

Il controllo della Rotazione

TPA_N gestisce la possibilità di applicare una rotazione alle parti. L'applicazione si differenzia in base alla modalità di ottimizzazione:
- *Square*: le parti possono ruotare di 90° in senso antiorario
- *True Shape*: le parti possono ruotare a passi di angoli, con possibilità di variazione del passo stesso.

Per ogni parte è possibile assegnare il grado di libertà consentito per la rotazione, con selezione possibile tra tre scelte:
- *0 (None)*: nessuna rotazione è ammessa per la parte
- *1 (90°)*: la parte ammette rotazione a passi di 90° (se ottimizza *True Shape*) o di 90° in senso antiorario (se ottimizza *Square*)
- *2 (Any)*: la parte ammette rotazione a passi variabili. Se ottimizza *Square*: la selezione corrisponde a quella precedente (*90°*).

L'assegnazione per le parti corrisponde al campo *Rotate* in struttura NestPart.

L'informazione relativa agli step di rotazione ammessi in selezione *Any* corrisponde all'assegnazione generale StepAngle:
- la proprietà ammette valori compresi tra 1.0 e 90.0

- il valore minimo di rotazione realmente applicata è ricondotto a un sottomultiplo intero di 360°.

Minore è il valore impostato e più impegnativa sarà la fase di calcolo dei piazzamenti, sia in termini di memoria richiesta che di tempo necessario a determinare una soluzione per i piazzamenti:
- impostare valore 45.0 per consentire rotazioni fino a passi minimi di 45.0°: una parte ha 8 possibili soluzioni di piazzamento, determinate applicando tutti i multipli di 45.0°, cioè: 0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°;
- impostare valore 15.0 per consentire rotazioni fino a passi minimi di 15.0°: una parte ha 24 possibili soluzioni di piazzamento, determinate applicando tutti i multipli di 15.0° (0°, 15°, 30°, …, 345°).

La lista delle rotazioni che sono applicate ad una determinata parte può comunque variare sulla base di valutazioni aggiuntive quali: dimensione e forma della parte, venatura assegnata.

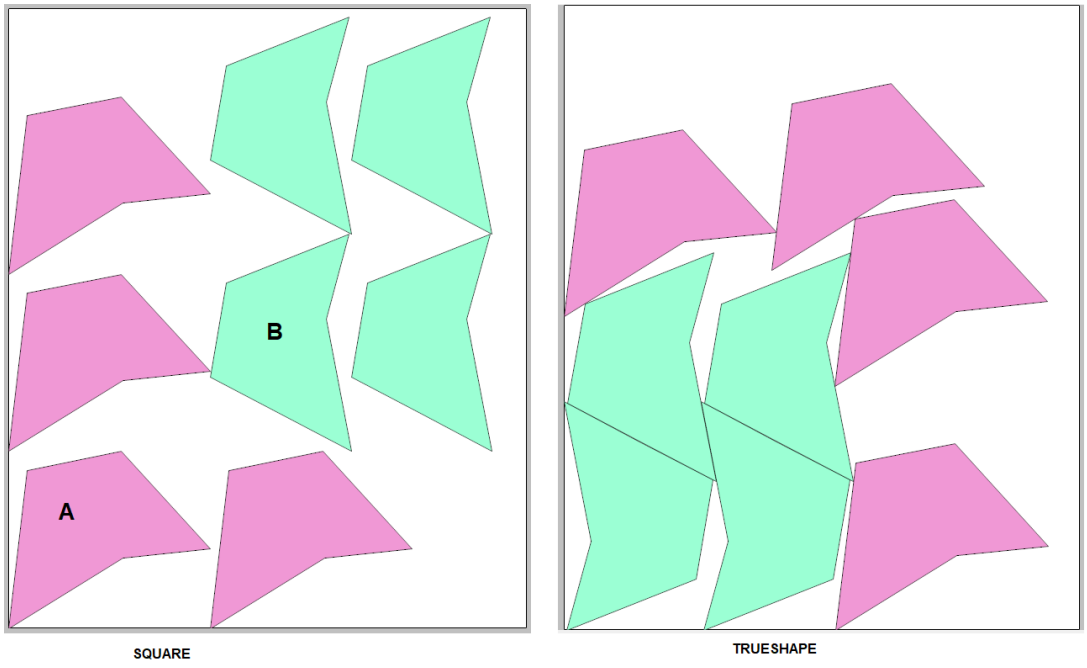
Minimizzazione del rettangolo di ingombro di una parte

Un ulteriore settaggio generale di tipo ***boolean*** può influenzare le logiche di rotazione delle parti (RctMinimize). L'abilitazione è applicabile alle parti per le quali non è assegnata una geometria rettangolare.

In caso di ottimizzazione *Square*:
- *false*: le parti sono utilizzate come assegnate. In caso di geometria assegnata: la stessa è piazzata come in originale oppure con rotazione di 90° (se consentita)
- *true*: il posizionamento iniziale della parte può essere preventivamente modificato, rispetto all'originale, con rotazione determinata in modo da minimizzare il proprio rettangolo di ingombro. Ciò può avvenire solo se la parte può ruotare.

In caso di ottimizzazione *True Shape*:
- *false*: se la parte ammette rotazione *any*, la determinazione della rotazione che minimizza il rettangolo di ingombro è comunque effettuata
- *true*: abilitazione analoga alla precedente anche per le parti che ammettono rotazione a passi di 90°.

La figura corrisponde all'esecuzione delle due ottimizzazioni con piazzamento di una stessa forma in ottimizzazione *Square* (a sinistra), *True Shape* (a destra) e con *RctMinimize=true*:
- **A**: la forma è utilizzata senza possibilità di rotazione. I piazzamenti corrispondono alla geometria originale
- **B**: la forma è utilizzata con possibilità di rotazione (90.0°). È ora evidente come ogni piazzamento parta da una rotazione che minimizza il rettangolo di ingombro della forma medesima, con conseguente maggiore possibilità di ottimizzazione dei piazzamenti



Rotazione e Venatura

L'assegnazione di venatura (X o Y) di una parte può limitarne le possibilità di rotazione o impedirne il piazzamento in lastre con venatura:
- è innanzitutto esclusa l'applicazione di rotazione (*Any*): se assegnata è ridotta a (*90°*)
- se parte e lastra hanno stessa venatura (es.: X+X; Y+Y): la parte può essere utilizzata con rotazioni (*0°*; *180°*) e solo (*0°*) se la parte non può ruotare
- se parte e lastra hanno differente venatura (es.: X+Y; Y+X): la parte può essere utilizzata solo se può ruotare e con rotazioni (*90°*; *270°*).

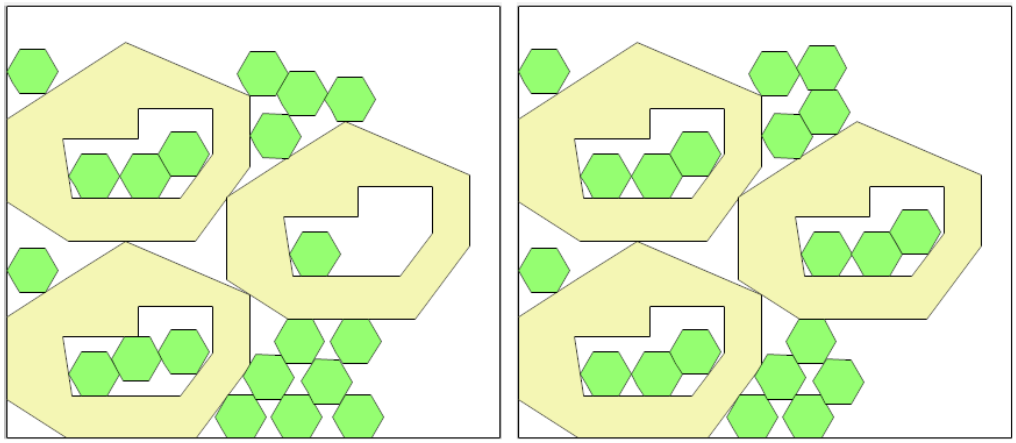
2.8 Il piazzamento nelle isole

Le *parti* possono assegnare isole utilizzabili per il piazzamento di altre parti più piccole.
La funzionalità è gestita solo in ottimizzazione *True Shape*.
Per ogni parte è possibile abilitare il piazzamento all'interno delle proprie isole, in corrispondenza al campo *UseIsle* in struttura NestPart.

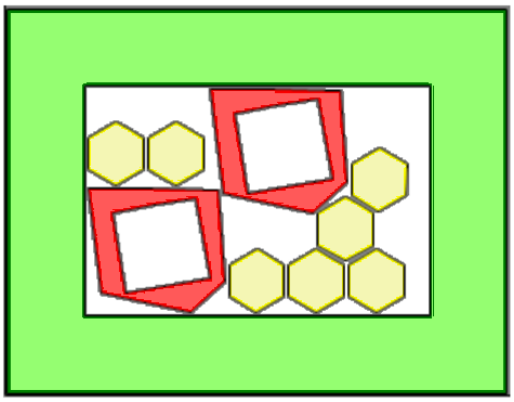
Alcuni settaggi generali di tipo ***boolean*** regolano i piazzamenti nelle isole:
- *PartInHole*: abilitazione complessiva ad effettuare piazzamenti entro le aree di sfrido delle parti
- *PartInHoleMulti*: abilitazione a effettuare piazzamenti ricorsivi entro le aree di sfrido delle parti
- *PartInHoleBefore*: abilitazione a privilegiare i piazzamenti entro le aree di sfrido delle parti.

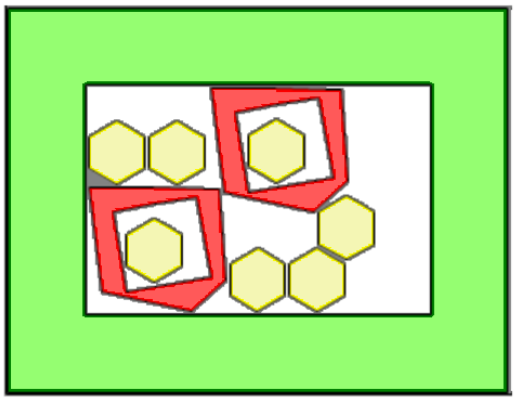
I termini della gestione di questa funzionalità devono essere valutati sulla base di considerazioni tecnologiche relative alla tenuta del foglio e al taglio delle parti.

La figura confronta l'effetto del settaggio *PartInHoleBefore*:
- a sinistra il caso di valore *false*: i piazzamenti sono effettuati secondo criteri generali di massimo riempimento della parte di foglio utilizzata
- a destra il caso di valore *true*: sono privilegiati i piazzamenti possibili entro le isole



La figura confronta l'effetto del settaggio *PartInHoleMulti*:
- a sinistra il caso di valore *false*: la parte riportata in colore *rosso* è posizionata nell'isola della parte di colore *verde*; entro la parte riportata in colore *rosso* non sono effettuati altri piazzamenti
- a destra il caso di valore *true*: piazzamenti sono effettuati anche nell'isola della parte riportata in colore rosso





2.9 Controllo del distanziamento tra i piazzamenti

Varie informazioni concorrono a determinare il distanziamento tra i piazzamenti e tra i piazzamenti e i bordi del foglio.

Diametro di taglio delle parti

Un'informazione generale assegna il diametro dell'utensile utilizzato per tagliare una parte: corrisponde al settaggio CutterDiameter.
Il valore del settaggio può essere nullo (=0.0) ed è utilizzato per le parti che non assegnano un proprio diametro utensile.
Per ogni parte è possibile assegnare un diametro utensile per il profilo primario ed uno per il taglio delle isole, in corrispondenza ai campi (*PartDiameter*, *IsleDiameter*) in struttura NestPart:

- se entrambi i campi hanno valore significativo (positivo): la parte utilizza specifici diametri per profilo primario ed isole
- *IsleDiameter=0.0*: la parte utilizza *PartDiameter* anche per i profili interni
- *PartDiameter=0.0, IsleDiameter=0.0*: la parte utilizza CutterDiameter sia per profilo primario che per le isole.

È anche possibile differenziare il diametro utilizzato per ogni profilo interno di una parte (si rimanda alla funzione IniGeometry). Se non altrimenti specificato, nel documento si ipotizza di utilizzare un'assegnazione unica per tutti i profili di una parte.

Di seguito si farà riferimento generico a *CutterDiameter* per indicare il diametro di taglio di una parte: nella realtà, il valore utilizzato tiene conto delle valutazioni qui riportate.

TPA_N supporta due modi per applicare la tecnologia ad una parte, in corrispondenza al settaggio *TecnoDistanceType* (di tipo ***boolean***).

Se *TecnoDistanceType=false*: l'utensile è applicato attorno al profilo di ogni parte:

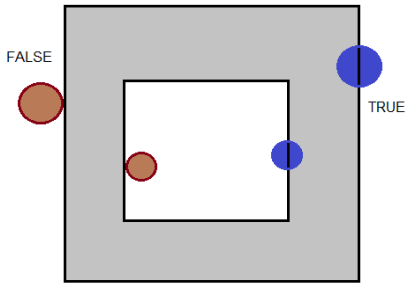
- sulla parte esterna del profilo esterno, ad aumentare l'area effettiva del profilo
- sulla parte interna dei profili di taglio degli sfridi, a diminuire l'area dello sfrido.

Se *TecnoDistanceType=true*: l'utensile è applicato a sormonto del profilo di ogni parte:

- l'area effettiva del profilo esterno è aumentata in misura della metà della dimensione dell'utensile
- l'area effettiva di uno sfrido è ridotta in modo analogo.

La figura riporta una parte con isola (entrambi i profili sono rettangolari):

- l'area delimitata tra i due profili è riportata in colore grigio
- i cerchi rappresentano l'utensile: per l'isola è utilizzato un utensile di diametro inferiore
- a sinistra i cerchi (di colore marrone) sono esterni ai profili, in corrispondenza a *TecnoDistanceType=false*
- a destra i cerchi (di colore blu) sormontano i profili, in corrispondenza a *TecnoDistanceType=true*



Se non altrimenti specificato, nel documento si ipotizza valida la situazione corrispondente a: *TecnoDistanceType=false.*

La figura riporta due parti generiche:
- a sinistra una parte senza isole
- a destra una parte con isola

La parte piena colorata delimita i percorsi assegnati per le parti: l'area netta di una parte.
I percorsi a tratti spesso delimitano l'ingombro effettivo delle parti, con aggiunto l'ingombro dell'utensile di taglio.

Tipologia dell'utensile di taglio delle parti

È possibile selezionare la tipologia dell'utensile di taglio, con l'impostazione del settaggio OneCutterDimension (tipo ***boolean***):
- *(false)* l'utensile ha dimensione di taglio in XY (Es.: fresa, laser)
- *(true)* l'utensile ha una sola dimensione di taglio (Es.: lama, taglierino).

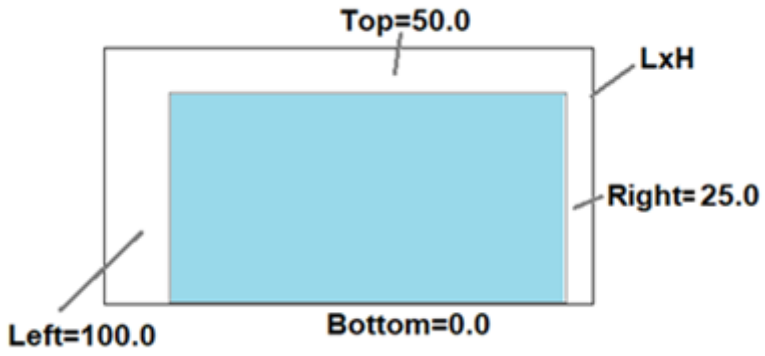
Nel secondo caso indicato, la dicitura *diametro dell'utensile* va interpretata più correttamente come *dimensione dell'utensile.*

L'impostazione è utilizzata in ottimizzazione *True Shape* come uno dei fattori che determinano il distanziamento tra piazzamenti contigui o tra le parti ed i bordi del foglio:
- l'utilizzo di utensile tipo (fresa, laser) introduce in automatico un distanziamento minimo pari all'*Errore cordale*, dato che il movimento dell'utensile attorno ad uno spigolo traccia un percorso esterno curvo, che deve essere approssimato in una serie di tratti lineari. In figura, la freccia indica il percorso esterno tracciato dall'utensile: ad ogni spigolo del rettangolo originale corrisponde lo sviluppo di un arco.

- l'utilizzo di utensile tipo (lama, taglierino) introduce un distanziamento minimo pari all'*Errore cordale* solo in considerazione di come sono assegnati i profili originali delle parti. Un esempio particolare può corrispondere a parti rettangolari o convesse, composte da soli tratti lineari e con taglio effettuato con un solo utensile: in questo caso il distanziamento tra le parti può non introdurre alcuna correzione aggiunta.

Distanziamenti dai bordi del foglio

Date le dimensioni di un *foglio* rettangolare, è possibile assegnare delle aree laterali non utili ai fini dei piazzamenti e differenziabili per lato. In figura un foglio di dimensioni LxH e con assegnazione dei quattro margini esterni tutti differenti: l'area utile per i piazzamenti è quella interna colorata.



L'impostazione generale dei margini esterni corrisponde ai settaggi:
- MarginOuter: per una assegnazione unificata dei quattro margini
- MarginLeft, MarginRight, MarginBottom, MarginTop: per una assegnazione differenziata dei margini.

Per ogni foglio è possibile assegnare un proprio margine esterno, in corrispondenza al campo (*Border*) in struttura NestSheet:

- *Border=0.0*: il foglio utilizza le impostazioni generali
- altrimenti (*Border>0.0*): il foglio utilizza il valore impostato in struttura.

In tutti i casi è possibile assegnare valore nullo o positivo: i margini possono solo diminuire l'area utilizzabile del foglio.

In caso di ottimizzazione *True Shape* e con foglio di forma non rettangolare: è utilizzato un margine esterno unico, assegnato al valore massimo impostato per i quattro margini differenziati per lato.

La *distanza minima* che può esserci tra il bordo di un foglio e un piazzamento, inteso come area netta, è di 0.0 mm. La distanza effettiva è determinata caso per caso, sulla base di molteplici aspetti:
- *CutterDiameter* è il diametro della tecnologia di taglio delle parti (il valore minimo utilizzabile è 0.0)
- il settaggio OverrunSheet
  - *false*: impone che l'utensile di taglio non esca dai margini del foglio, portando quindi la distanza minima dai bordi al valore (*CutterDiameter*)
  - *true*: permette all'utensile di uscire dai margini del foglio, in base al settaggio MaximizeOverrunSheet (per intero o per metà del suo valore)
- la valutazione di distanza minima dei piazzamenti dai bordi può aggiungere un distanziamento a compensazione di:
  - margini esterni insufficienti
  - modifica dei profili con applicazione dell'*errore cordale*
- In caso di ottimizzazione *True Shape* e se sono assegnati diametri di tecnologia differenti per le parti, il diametro da considerarsi ai punti precedenti è quello minimo. Ciò comporta che parti differenti possono accostarsi al bordo del foglio in modo differente: maggiore è il diametro di taglio della parte e maggiore sarà la distanza minima dal bordo.

Distanziamenti dalle aree di vincolo interne al foglio

Il/i margini come assegnati nel paragrafo precedente non sono applicati alle aree di vincolo interne al foglio.
In caso di ottimizzazione Square, le aree di vincolo sono considerate per il rettangolo di ingombro dei profili che le definiscono, di fatto aumentando le aree stesse.

Le aree di vincolo definiscono zone dove non è possibile effettuare piazzamenti.
Per ogni foglio è possibile assegnare due campi relativi all'utilizzo delle aree di sfrido (*IsleBorder*, *IsleDiameter*) in struttura NestSheet. Per entrambi i campi sono significativi valori positivi o nulli. L'utilità dei campi può essere molteplice ed è possibile distinguere tra due situazioni principali:

- necessità di assegnare un distanziamento minimo dei piazzamenti dalle aree di vincolo della lastra
- utilizzo delle aree di vincolo della lastra per assegnare piazzamenti preliminari, anche derivanti da altro software di nesting e gestiti con proprie logiche di funzionamento (vedi: Nesting incrementale).

Le due situazioni possono coesistere nell'assegnazione di una lastra.

Vediamo ora come formalizzare le diverse situazioni:

1. *(IsleBorder <=0.0, IsleDiameter<=0)*: nessun margine è applicato tra le aree di sfrido ed il taglio delle parti (per le aree di vincolo il settaggio OverrunSheet è sempre applicato con valore *false,* escludendo la possibilità di sovrapposizione con il taglio delle parti)

2. *(IsleBorder >0.0, IsleDiameter<=0)*: per le aree di vincolo è assegnato un margine di rispetto esterno (vedi caso **A** di figura). Anche in questo caso, il settaggio OverrunSheet è applicato con valore *false*.

3. *(IsleBorder >0.0, IsleDiameter >0)*: l'area di vincolo può ora corrispondere ad un piazzamento già presente sulla lastra
   - *IsleBorder* posiziona l'ingombro esterno dell'utensile di taglio dell'area di vincolo
   - *IsleDiameter* assegna la tecnologia associata all'area di vincolo.

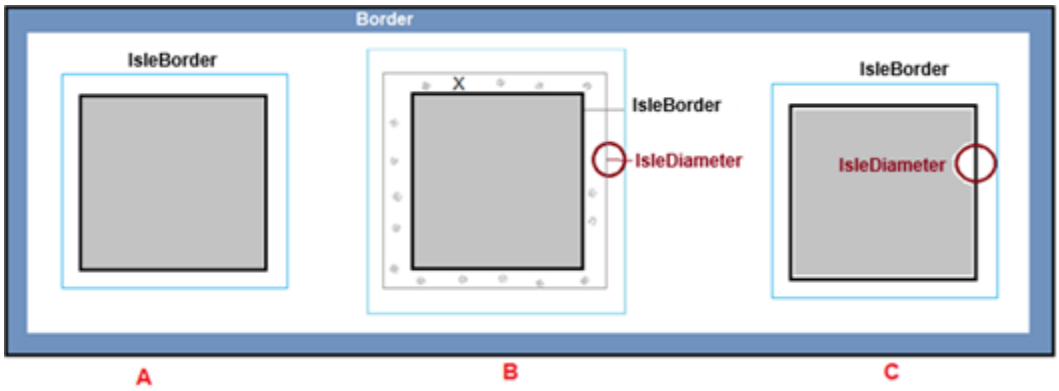
   Si possono qui differenziare due situazioni:
   - *IsleBorder >= IsleDiameter*: il taglio è esterno al profilo assegnato per l'area di vincolo (vedi caso **B** di figura)
   - *IsleBorder < IsleDiameter:* il taglio è parzialmente interno al profilo assegnato per l'area di vincolo. È gestita la condizione limite corrispondente a *IsleBorder= IsleDiameter*0.5* (il centro dell'utensile è lungo il profilo dell'area: vedi caso **C** di figura).

   L'area di vincolo è ora gestita al pari di una parte già piazzata:
   - applica valutazioni per sovrapposizioni e distanziamenti con profili di taglio di altri piazzamenti (settaggi: OverrunPolygon, MarginInner, OverrunSecurity)
   - non applica invece il settaggio TecnoDistanceType
   - non applica il settaggio OverrunSheet
   - l'assegnazione di parametri differenziati per ingombro e tecnologia corriponde al fatto che piazzamenti già assegnati possono derivare da altro software di nesting, funzionante con proprie logiche, e la gestione di due parametri garantisce il massimo della flessibilità.

4. da ultimo il caso *(IsleBorder <=0.0, IsleDiameter >0)*: assume *IsleBorder = IsleDiameter*, con soluzione di profilo di taglio esterno all'isola.

La figura riporta una lastra rettangolare con:

- bordo esterno aggiunto (margine di colore blu)
- tre aree di sfrido con indicati i campi (*IsleBorder*, *IsleDiameter*). Ogni area è contrassegnata con una lettera (A, B, C)

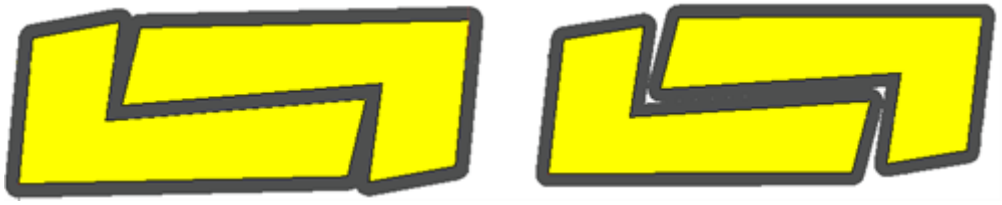


È anche possibile differenziare margine e diametro utilizzati per ogni area interna ad un foglio (si rimanda alla funzione IniGeometry). Se non altrimenti specificato, nel documento si ipotizza di utilizzare un'assegnazione unica per tutti i profili interni di un foglio.

Distanziamento tra i piazzamenti

La *distanza minima* che può esserci tra piazzamenti contigui è pari a (CutterDiameter + OverrunSecurity):

- il settaggio OverrunPolygon gestisce l'abilitazione di sormonto delle aree di taglio delle parti. In figura un esempio di piazzamenti con sormonto (a sinistra) e senza sormonto (a destra): il contorno in colore grigio corrisponde al percorso dell'utensile di taglio



- il settaggio OverrunSecurity assegna la *Distanza di sicurezza* aggiunta ai piazzamenti con applicato il sormonto delle aree di taglio delle parti
- il settaggio MarginInner assegna un distanziamento aggiunto tra piazzamenti contigui.

Vediamo alcuni esempi di impostazioni possibili

CutterDiameter	*OverrunPolygon*	*OverrunSecurity*	*MarginInner*	**Distanza tra piazzamenti**
10.0	False	0.0	0.0	20.0
10.0	False	0.0	20.0	40.0
10.0	True	0.5	0.0	10.5
10.0	True	0.0	20	30.0

In caso di ottimizzazione *True Shape*, la tabella è semplificata rispetto all'assegnazione di differenti diametri per le parti o di geometrie comunque variabili. In tal caso:

- il massimo sormonto tra profili di taglio è pari alla metà del diametro inferiore assegnato per le parti (tutte quelle piazzabili)
- la distanza minima tra le parti può aggiungere un distanziamento a compensazione di modifica di tratti di profili con applicazione dell'*Errore cordale*.

Ingombri esterni alle parti

Per le parti è possibile assegnare quattro campi interpretati come ingombri esterni differenziati per lato, da aggiungere al profilo della parte:

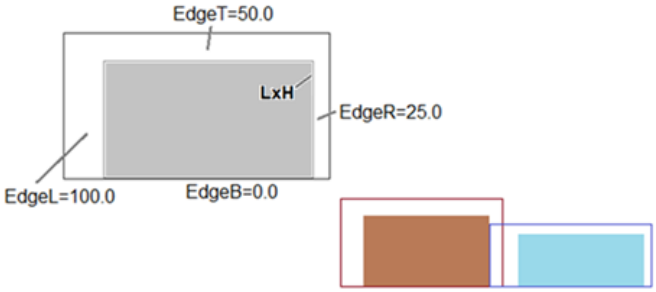
- le assegnazioni sono in campi (*EdgeL*, *EdgeR*, *EdgeB*, *EdgeT*) in struttura NestPart
  - ogni valore assegna un ingombro lineare misurato lungo un asse coordinato (X oppure Y)
- questi ingombri assegnano aree considerate di scarto:
  - ingombri esterni contigui possono essere in sovrapposizione

- in tutti gli ingombri esterni sono esclusi piazzamenti utili.

In ottimizzazione *True Shape* gli ingombri esterni sono generalmente applicati in eccesso, rispetto all'ottimizzazione *Square*:
- se la parte assegna una geometria non rettangolare: si utilizza un unico valore, pari all'ingombro massimo impostato.
- gli ingombri aumentano il distanziamento delle parti dai bordi del foglio.

L'utilizzo degli ingombri esterni è da ritenersi una particolarità di applicazioni che richiedono l'ottimizzazione *Square* di parti tipicamente rettangolari.

Sulla parte superiore della figura è riportata una parte (individuata tramite le dimensioni esterne del profilo LxH) con assegnati ingombri esterni tutti differenti.
Sulla parte inferiore un esempio di massimo avvicinamento orizzontale tra due parti con i relativi margini esterni



In caso di simmetria e/o rotazione di una parte, i margini *seguono* le trasformate applicate. Poniamo il caso di parte con applicato speculare X:
- i campi (*EdgeL*, *EdgeR*) sono applicati scambiati.

Il significato degli ingombri esterni è strettamente tecnologico e dipende dall'applicazione specifica della libreria TPA_N. Un esempio può riguardare l'esecuzione di lavorazioni di pertinenza della singola parte piazzabile e che richiedono l'utilizzo di spazi esterni alla geometria della parte medesima.
Come già è stato detto, l'utilizzo prevalente è da ritenersi una particolarità di applicazioni che richiedono l'ottimizzazione Square di parti tipicamente rettangolari o assimilabili a rettangoli. Nel caso in cui fosse necessario discriminare l'utilizzo degli ingombri in funzione dell'ottimizzazione richiesta, è responsabilità del client gestire la situazione.

2.10 Abbinamenti di parti

TPA_N gestisce la possibilità di attivare raggruppamenti di parti da utilizzarsi per i piazzamenti anziché per le parti singole.

Abbinamento automatico

Un *abbinamento* (o anche *cluster*) automatico consiste nella generazione di un gruppo ottenuto per una singola parte con accoppiamento della parte con una copia di sé stessa ruotata di 180°.
La funzionalità è gestita solo in ottimizzazione *True Shape*.

La figura riporta un esempio di parte singola sulla parte sinistra e, a destra, il *gruppo* ottenibile con applicazione dell'abbinamento automatico

Il piazzamento di una parte in modalità di *Abbinamento automatico* viene privilegiato rispetto al piazzamento singolo se l'efficienza di utilizzo del *gruppo* è almeno uguale al valore assegnato con proprietà *ClustersExpected*, dove l'efficienza di utilizzo di un abbinamento automatico è calcolata come:

(Area della parte singola * 2 * 100) / (Lunghezza gruppo * Altezza gruppo)

Per ogni parte è possibile richiedere il piazzamento privilegiato di abbinamento automatico in corrispondenza al campo *AutoPair* in struttura NestPart. Il meccanismo di abbinamento automatico richiede che la parte sia ruotabile.

Se possibile TPA_N ricerca l'abbinamento automatico più efficiente applicando la rotazione che minimizza l'ingombro della parte.
Il piazzamento di un abbinamento automatico può applicare ulteriori rotazioni del *gruppo*, come primo tentativo a passi di 90° e successivi a passi inferiori (45°, 30°, ...).
La parte può anche essere piazzata singolarmente, laddove non risultasse più possibile applicarla in *gruppo*.

Il settaggio *AutomaticCluster* assegna l'abilitazione complessiva a effettuare abbinamenti automatici.

La proprietà *ClustersAbsolute* assegna l'efficienza di utilizzo di un abbinamento automatico in modo autonomo per le parti interessate ed indipendente dallo stesso settaggio *AutomaticCluster*: se un abbinamento automatico ha un'efficienza almeno uguale al valore impostato, il piazzamento in abbinamento è attivato comunque.

L'applicazione in abbinamento automatico può essere applicata forzatamente attiva o no in caso di riconoscimento di profili di forme notevoli. Ad esempio:
- non applica in caso di: rettangolo, cerchio, ellisse, alcune tipologie di poligoni regolari
- forza l'applicazione in alcune tipologie di poligoni regolari.

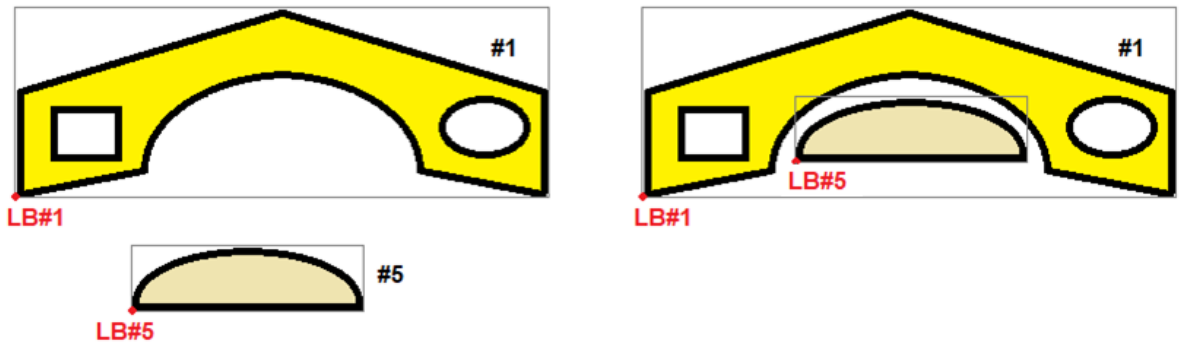
Abbinamento manuale

Il meccanismo permette di creare raggruppamenti tra singole *parti* assegnate in modo indipendente: le parti così raggruppate saranno posizionate come un blocco unico, mantenendo invariata la posizione reciproca tra le parti.
In caso di abbinamento manuale tra parti, TPA_N costruisce un disegno complessivo delle varie parti, comprensivo delle eventuali isole: il disegno così ottenuto è poi utilizzato come una entità unica da piazzare sui fogli.

L'assegnazione di abbinamenti manuali è gestita sia in ottimizzazione *True Shape* che *Square*, con le differenze proprie relative alle due procedure specifiche.
Di seguito, il termine *cluster* sarà utilizzato per individuare un generico abbinamento manuale.

In figura sono rappresentate due parti:
- #1 è la parte con identificativo 1: profilo primario e 2 isole
- #5 è la parte con identificativo 5: un unico profilo primario.

Per entrambe le parti è indicato il vertice LB (inferiore sinistro) del rettangolo di ingombro del profilo primario.
A destra le parti sono riportate su un medesimo piano di rappresentazione, che evidenzia il posizionamento reciproco



La costruzione corretta di un cluster deriva dall'assegnazione corretta delle parti. Ad esempio: è responsabilità dell'applicazione chiamante assegnare le parti in modo che abbiamo intersezioni oppure che siano geometricamente distinte.
Le parti realmente prese in considerazione sono solo quelle abilitate (struttura NestPart, campo *Enable=true*).

Un abbinamento manuale è trattato dalla procedura di nesting coma una parte singola: il piazzamento del cluster non è opzionale ma richiesto.
A tutti gli effetti, un cluster è da intendere simile ad una parte solo con una modalità di costruzione geometrica differente.
Le considerazioni fino a qui fatte in merito all'assegnazione ed utilizzo di parti singole sono estendibili ai cluster (Es.: controllo della venatura, applicazione di trasformate, piazzamento in abbinamento automatico): questo vale anche per le considerazioni che seguiranno, se non altrimenti specificato.

Un abbinamento manuale è assegnato tramite appositi metodi e strutture.
L'utilizzo di una parte in composizione di uno o più cluster non esclude la possibilità di ottenere piazzamenti diretti per la stessa: parliamo in questo caso di *piazzamenti residui*.

In analogia a quanto vale per una parte, un cluster può assegnare un insieme di informazioni generali:
- quantità richiesta e massima disponibile
- trasformate ammesse e richieste (rotazione, speculare)
- assegnazioni di corrispondenza (spessore, materiale, colore, venatura)
- assegnazioni di abilitazioni (cluster automatico, piazzamento in isole, piazzamento a griglia)
- priorità.

La costruzione geometrica del cluster avviene indicando quali parti concorrono alla formazione dello schema del cluster:
- sono utilizzabili tutte le parti assegnate
- ogni parte può essere utilizzata una o più volte, nella costruzione di un cluster o di cluster diversi
- un cluster può indicare fino a un massimo di 100 utilizzi distinti di parti
- ogni utilizzo di parte assegna le trasformate da applicare: traslazione, speculare, rotazione.

Gruppi e filtri di corrispondenza

Quanto già detto in merito alle condizioni di corrispondenza e/o di filtri tra *parti* e *fogli* è valido anche tra *cluster* e *fogli*.
Nessuna corrispondenza è invece richiesta tra un *cluster* e le *parti* in esso utilizzate, con la sola eccezione riguardante la venatura. Vediamo cosa ciò comporta:

- si assegna un cluster con spessore pari a 20 mm: ogni parte potrà essere utilizzata nella costruzione del cluster, a prescindere dallo spessore della parte. In particolare: può essere utilizzata anche una parte che non verifica alcuna corrispondenza con i fogli
- si assegna un cluster con venatura X: potrà essere utilizzata una parte solo se assegnata con venatura diversa da Y
- si assegna un cluster con venatura Y: potrà essere utilizzata una parte solo se assegnata con venatura diversa da X
- si assegna un cluster con venatura OFF: potrà essere utilizzata una parte solo se assegnata con venatura OFF.

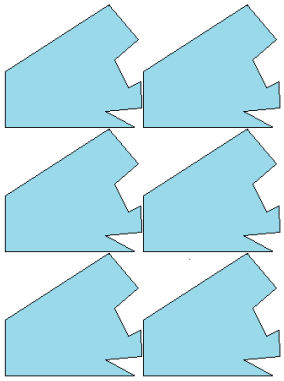
Piazzamenti a matrice

Per una parte è possibile richiedere piazzamenti secondo uno schema a matrice, in corrispondenza al campo *AutoGrid* in struttura NestPart.

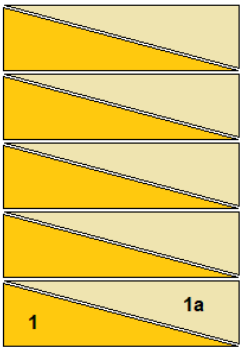
Il settaggio *AutomaticGrid* assegna l'abilitazione complessiva a effettuare piazzamenti a matrice.
La funzionalità è gestita con maggiore efficacia in ottimizzazione *True Shape*: di seguito è indicata come prerogativa dell'ottimizzazione *True Shape*, ma è ora parzialmente funzionante anche in ottimizzazione *Square*.

I pezzi per i quali è richiesto un piazzamento a matrice sono utilizzati prima degli altri e vengono piazzati con disposizione *riga* * *colonna*, in considerazione dello spazio disponibile sul foglio. Al fine di ottimizzare i piazzamenti, ogni parte può essere analizzata applicando autonome strategie di abbinamento:
- una parte può essere piazzata con ripetizione di una unità che può corrispondere a una parte singola, ripetuta sempre con la stessa rotazione, o a due parti, con applicazione di abbinamento automatico
- l'unità di ripetizione, di parte singola o doppia, può poi essere posizionata valutando una variazione di rotazione di 0° o 90°.

Di seguito sono illustrati esempi di piazzamento a matrice di una parte



- non è applicato abbinamento automatico
- lo sviluppo a matrice è di: 3 righe * 2 colonne



- è applicato l'abbinamento automatico della parte, costituito dall'accoppiamento delle parti (**1;1a**)
- lo sviluppo a matrice è di: 5 righe * 1 colonna

In ottimizzazione *True Shape*, l'applicazione di piazzamenti a matrice può essere applicata forzatamente attiva o no in caso di riconoscimento di profili di forme notevoli.
In ottimizzazione *Square*, il piazzamento a matrice è limitato al solo primo tipo di elemento piazzato.

Raggruppamenti di parti

Per illustrare questa funzionalità, supponiamo che le parti di un progetto corrispondano a differenti prodotti finali. Naturalmente si desidera che tutte le parti di un prodotto siano piazzate raggruppate su un foglio o su fogli successivi. È questa la situazione tipica in cui è possibile utilizzare la funzionalità *Raggruppamento di parti*.

Per la funzionalità sono predisposti il campo *Range* in strutture NestPart e NestCluster e i settaggi UseRangePart e DimRangePart.

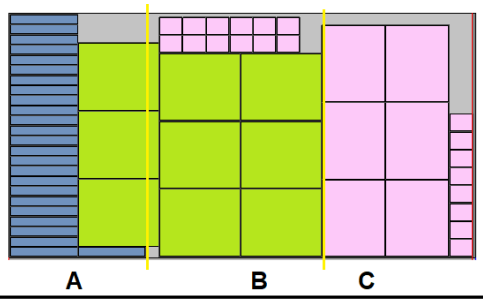
Parti e/o cluster con stesso valore *Range* costituiscono un gruppo ed i piazzamenti sono effettuati con suddivisione a zone dell'area utile, in modo da raggruppare i pezzi di un gruppo:

- gli elementi piazzabili sono ordinati per valori crescenti di raggruppamento (0, 1, … fino al valore massimo)
- l'ordine corrisponde all'ordine di piazzamento
- non c'è garanzia che tutte le parti di un gruppo siano piazzate su un unico foglio.

Il settaggio UseRangePart assegna l'abilitazione complessiva alla funzionalità, con selezione possibile fra tre scelte:

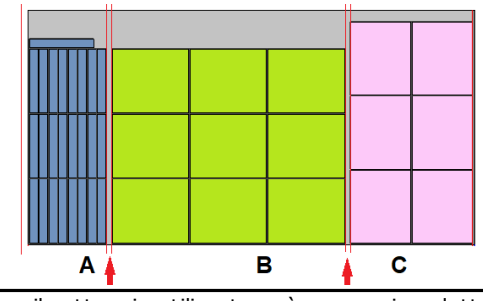
- *0*: la funzionalità è disabilitata (disattiva l'interpretazione del campo *Range)*
- *1*: richiede piazzamenti in aree continue
  - i piazzamenti di un gruppo sono consecutivi, ma non separati da quelli del gruppo precedente, mischiandosi però al massimo con i piazzamenti del gruppo precedente
  - le aree interessate ai piazzamenti di gruppi consecutivi sono tra loro contigue, ma senza una separazione netta. La contiguità delle aree rispetta la direzione selezionata per il nesting
    - *direzione orizzontale*: le aree si accostano in direzione orizzontale

- *direzione verticale*: le aree si accostano in direzione verticale



La figura illustra la funzionalità:
- **A**: estremo di area con elementi di range =0 (parti di colore: blu)
- **B**: estremo di area con elementi di range successivo (=1; parti di colore: verde)
- **C**: estremo di area con elementi di range successivo al precedente (=2; parti di colore: fucsia): parti del range possono essere piazzate anche in area B ma non **A**

- *2*: richiede piazzamenti in aree separate
  - i piazzamenti di un gruppo sono in area contigua e separata da quella del gruppo precedente. Il settaggio DimRangePart imposta la spaziatura tra aree vicine.



La figura illustra la funzionalità:
- sono indicati due canali di separazione tra aree: qui i canali sono verticali, in corrispondenza a direzione di nesting orizzontale
- è normale ipotizzare che i canali di separazione corrispondano a successivi tagli della lastra

  - il settaggio utilizzato può essere ricondotto al caso precedente in caso di ottimizzazione True Shape ed utilizzo di lastra non strettamente rettangolare. In questo caso, è infatti possibile rispettare uno schema di piazzamenti in aree separate solo se la lastra è considerata assimilabile ad un rettangolo. Vediamo le condizioni che verificano una condizione di simil-rettangolo per una lastra generica:
    - 1 sola area primaria (cioè: il profilo originale non ha intersezioni o strozzature estreme)
    - non assegna isole
    - se non riconosce forma nota (rettangolo, ellisse, cerchio), deve essere convessa ed il rapporto tra area complessiva e area del rettangolo di ingombro della lastra deve valere almeno 0.75.

Il settaggio UseRangePart è applicato solo se un progetto richiede valori differenti di campo Range.

Il raggruppamento tra elementi è applicato solo a parti e/o cluster con quantità richiesta di utilizzo strettamente positiva (campo *N* in strutture NestPart e NestCluster): sono cioè esclusi elementi con soli piazzamenti extra (vedi paragrafo: Piazzamenti a riempimento).

Un raggruppamento di piazzamenti include anche eventuali piazzamenti extra richiesti per gli elementi di un gruppo (vedi paragrafo I piazzamenti extra):

- i piazzamenti extra di un gruppo possono riempire una lastra, ma non iniziare la successiva: non è quindi garantito l'utilizzo totale delle quantità extra di un gruppo
- il loro utilizzo non valuta il settaggio ExtraFiller.

Eventuali piazzamenti a riempimento sono valutati a valle di tutto e interessano l'intera area della lastra, ma con possibili differenze in base al valore di *UseRangePart* applicato:

- in caso di piazzamenti in aree continue: i piazzamenti sono piazzati ovunque
- in caso di piazzamenti in aree separate: i piazzamenti rispettano i canali di separazione tra le aree.

2.11 Ripetizioni di parti e fogli

La normale condizione di assegnazione delle *parti* e dei *fogli* corrisponde ad avere liste con più elementi abilitati (campo *Enable* in strutture NestPart e NestSheet).
In questa situazione: sono presi in considerazione solo gli elementi che assegnano una quantità richiesta/disponibile strettamente positiva (campo *N* in strutture *NestPart* e *NestSheet*).

Tutto quanto diciamo si intende riferito ad un solo gruppo di corrispondenza tra fogli e parti: se i gruppi sono più di uno, le considerazioni vanno applicate ad ogni gruppo singolo.
Quanto detto per l'utilizzo delle parti può in parte differire per il piazzamento dei cluster, per i quali si rimanda al paragrafo Piazzamenti di cluster.

La procedura di ottimizzazione piazza i pezzi sul minor numero di pannelli disponibili. Se la procedura ha piazzato la quantità disponibile dei pezzi e per alcuni di essi è impostata una **Quantità massima > Quantità richiesta**, il piazzamento riempie i pannelli già utilizzati, fino al valore massimo impostato.
Per la gestione del campo *Quantità massima di una parte*, si rimanda all'argomento I piazzamenti extra.

Indichiamo questa situazione con dicitura: *Parti multiple* e *Fogli multipli*.
Esaminiamo di seguito le situazioni particolari che è possibile derivare da questa generale.

Come sarà evidente dai prossimi paragrafi, molte sono le proprietà che possono influenzare il riconoscimento di situazioni specifiche, con possibili differenti interpretazioni attribuibili alle quantità richieste per le parti.

Nel caso in cui l'interpretazione di queste situazioni non risulti intuitiva, si consiglia di escludere in modo esplicito l'utilizzo di una parte, piuttosto che azzerarne la quantità richiesta.

Parte singola e Foglio singolo

La lista delle *parti* e dei *fogli* hanno entrambe un solo elemento abilitato. In base alla necessità è possibile selezionare tra ottimizzazioni specifiche:

NestPart.N	*NestSheet.N*	
0	0	la procedura piazza il maggior numero di pezzi su 1 foglio
0	>0	la procedura piazza il maggior numero di pezzi sul totale disponibile del foglio (tutti i fogli risultanti sono uguali)
>0	0	la procedura calcola il numero di fogli necessari a piazzare la Quantità richiesta della parte. Se per la parte è impostata una *Quantità massima > Quantità richiesta*, l'ultimo foglio utilizzato è riempito fino al valore massimo impostato
>0	>0	la procedura piazza la Quantità richiesta della parte sul minor numero di fogli disponibili. Se la procedura ha piazzato la quantità disponibile e per la parte è impostata una *Quantità massima > Quantità richiesta*, l'ultimo foglio utilizzato è riempito fino al valore massimo impostato

Parti multiple e Foglio singolo

La lista delle *parti* ha più di un elemento abilitato, mentre quella dei *fogli* solo uno. La *Quantità disponibile delle parti* deve essere strettamente positiva (> 0).
In base alla necessità è possibile selezionare tra ottimizzazioni specifiche:

NestSheet.N	
0	la procedura calcola il numero di fogli necessari a piazzare il totale delle parti richieste
>0	la procedura piazza il massimo delle parti richieste sul minor numero di fogli disponibili

In entrambi i casi: se la procedura ha piazzato la quantità disponibile dei pezzi e per alcuni è impostata una *Quantità massima > Quantità richiesta*, il piazzamento riempie i fogli già utilizzati fino al valore massimo impostato.

Parte singola e Foglio multiplo

La lista delle *parti* ha un solo elemento abilitato, mentre quella dei *fogli* più di uno. La *Quantità disponibile dei fogli* deve essere strettamente positiva (> 0).
In base alla necessità è possibile selezionare tra ottimizzazioni specifiche:

NestPart.N	
0	la procedura piazza il maggior numero di pezzi nei fogli disponibili
>0	la procedura piazza il numero richiesto per la parte sul minor numero di fogli disponibili. Se la procedura ha piazzato la quantità disponibile e per la parte è impostata una *Quantità massima > Quantità richiesta*, il piazzamento riempie l'ultimo foglio già utilizzato, fino al valore massimo impostato

I piazzamenti extra

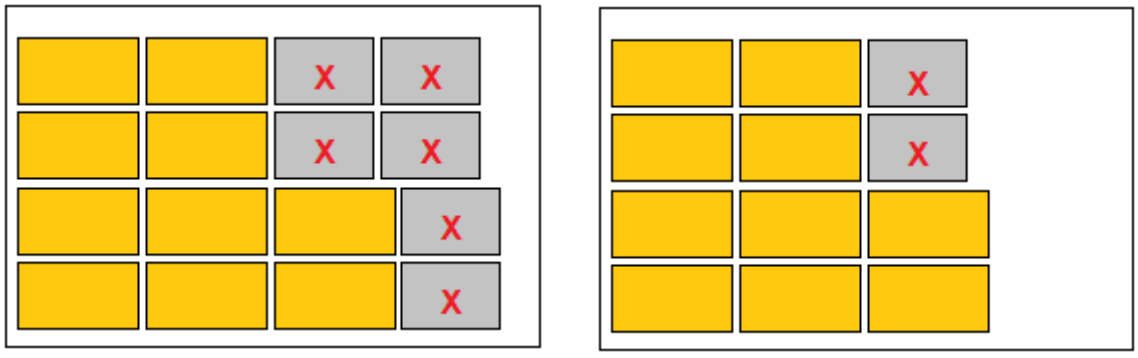
Per le parti e i cluster è possibile assegnare dei piazzamenti extra, in aggiunta a quelli effettivamente richiesti.

L'informazione è assegnata in campo *Nmax* di struttura NestPart. Il valore è utilizzato se:
- per i piazzamenti richiesti (in campo *N* della struttura) è assegnato valore strettamente positivo, oppure
- risulta *UseOnlyExtraPart*=true

Il significato attribuito al valore è determinato dal settaggio *ModeExtraPart (di tipo **boolean**)*:
- se è *ModeExtraPart=false (default):* il valore impostato è maggiore di quello dei piazzamenti richiesti. In queste condizioni, il numero dei *piazzamenti extra* è calcolato pari a: *(Nmax - N)*.
- se è *ModeExtraPart=true*: il valore impostato è positivo ed assegna direttamente il numero dei piazzamenti extra.

Se non altrimenti specificato, nel documento si ipotizza valida la situazione corrispondente a: *ModeExtraPart*=false, *UseOnlyExtraPart*=false
Su un foglio, i piazzamenti extra sono utilizzati solo dopo verifica di non potere più piazzare parti richieste.
In base a quanto detto risulta evidente come in nessun caso un foglio possa essere utilizzato solo per applicare piazzamenti extra.
Nell'utilizzo dei piazzamenti extra è possibile scegliere tra due modalità, come da settaggio ExtraFiller:
- *False*: i piazzamenti extra sono applicati a riempimento dei fogli
- *True*: i piazzamenti extra sono applicati solo a riempimento della lunghezza o altezza già impegnata con i pezzi richiesti. La direzione di riempimento è scelta in base alla direzione di avanzamento per i piazzamenti (vedi settaggio Direction):
  - se Orizzontale: il riempimento è applicato alla lunghezza del foglio, mentre non è applicato limite in altezza
  - se Verticale: il riempimento è applicato all'altezza del foglio, mentre non è applicato limite in lunghezza.

La figura illustra il risultato ottenuto con differente valore della proprietà:
- l'ipotesi di lavoro corrisponde a: *Direction*=0, *Corner*=0
- i piazzamenti extra sono segnati con una 'X'
- a sinistra il caso di *ExtraFiller = False*
- a destra il caso di *ExtraFiller = True*



A volte è richiesto di applicare il settaggio ExtraFiller solo sull'ultimo foglio e non su tutti i fogli. Questo può essere ottenuto utilizzando la funzione *ExtraFillerLastSheet*.

Piazzamenti a riempimento

Vediamo alcune considerazioni per le parti con richiesta di soli piazzamenti extra (in struttura NestPart: *N* =0, Nmax >0), anche indicati con la dicitura di *piazzamenti a riempimento.*
L'utilizzo delle parti è possibile solo se risulta *UseOnlyExtraPart*=true.

Le parti interessate sono sempre utilizzate *a valle di tutti gli elementi piazzabili*:
- l'ordine reciproco di utilizzo delle parti rispetta i normali criteri di ordinamento (priorità, dimensioni, tipologia, quantità, …)
- l'utilizzo delle parti applica le proprietà assegnate a riempimento delle lastre (*ExtraFiller*, *ExtraFillerLastSheet*)

L'utilità principale può essere di aggiungere piazzamenti ad una soluzione di nesting, allo scopo di ridurre gli sfridi. Un client può ad esempio assegnare le parti interessate in una lista separata utilizzabile a richiesta a *riempimento* di una soluzione di nesting.

Piazzamenti di abbinamenti manuali (cluster)

Qualche considerazione aggiuntiva è necessaria per l'utilizzo di elementi in abbinamento manuale (cluster). Le assegnazioni di quantità richiesta sono ora in struttura *NestCluster*, campi *N* e *Nmax*:
- per i cluster è esclusa la gestione di quantità richiesta nulla (campo *N*=0): in tal caso, il cluster è sempre automaticamente escluso

- se tutte le parti utilizzate in un cluster assegnano quantità disponibile nulla (campo *N* di struttura *NestPart* è uguale a 0)
  - le quantità piazzabili per il cluster sono quelle assegnate in campi (*N* e *Nmax*) di struttura *NestCluster*
  - le parti utilizzate nella costruzione del cluster non potranno essere utilizzate per piazzamenti diretti

- altrimenti: le quantità (*N* e *Nmax*) assegnate per il cluster sono confrontate con quelle delle singole parti
  - le quantità piazzabili per il cluster sono limitate dalle quantità delle parti, con possibilità estrema di escludere l'applicazione del cluster medesimo
  - se una parte è disponibile con quantità sufficiente, è possibile che la stessa sia poi utilizzata per piazzamenti diretti.

Nella valutazione di quanto detto è necessario tenere presenti alcuni aspetti complessivi:
- una parte può essere utilizzata più volte in un cluster ed anche in cluster differenti
- la lista dei cluster è scandita in sequenza di identificativo primario ed ognuno *prenota* la quantità richiesta per ogni parte: ciò può determinare insufficienza di quantità per un cluster successivo.

Un utilizzo particolare dei cluster può corrispondere a quanto abbiamo già riportato: assegnare quantità disponibile nulla (=0) per tutte le parti utilizzate nei cluster. La lista delle parti assegna di fatto una libreria di forme disponibili, mentre il vero progetto di nesting è assegnato direttamente sui cluster.

Spieghiamo con qualche esempio.

A. Si assegna un cluster con quantità richieste: *N=10*, *Nmax=20*. Il cluster utilizza 3 parti:
   - parte con ID=2, per la quale risultano: *N=5*, *Nmax=10*
   - parte con ID=3, per la quale risultano: *N=7*, *Nmax=10*
   - parte con ID=5, per la quale risultano: *N=5*, *Nmax=15*

   - le quantità piazzabili per il cluster risultano: *N=5*, *Nmax=10*
   - la parte con ID=2 non può essere utilizzata per piazzamenti diretti
   - la parte con ID=3 può essere utilizzata per piazzamenti diretti in quantità: *N=2*, *Nmax=0*
   - la parte con ID=5 può essere utilizzata per piazzamenti diretti solo di tipo extra (*Nmax=5*).

B. Si assegna lo stesso cluster dell'esempio precedente, ma con quantità cambiate per le parti utilizzate
   - parte con ID=2, per la quale risultano: *N=0*
   - parte con ID=3, per la quale risultano: *N=7*, *Nmax=10*
   - parte con ID=5, per la quale risultano: *N=5*, *Nmax=15*

   - il cluster non può essere piazzato

- la parte con ID=2 può essere utilizzata solo come parte singola in un gruppo di corrispondenza con i fogli
- le parti restanti possono essere utilizzate per piazzamenti diretti in base alle quantità impostate.

C. Si assegna lo stesso cluster dell'esempio precedente, ma con quantità per le parti tutte nulle:
- le quantità piazzabili per il cluster risultano quelle richieste: *N=10*, *Nmax=20*
- le parti non sono utilizzabili per piazzamenti diretti.

In caso di piazzamenti residui risultanti delle parti utilizzate in cluster, ogni parte è utilizzata in base al gruppo di corrispondenza che la parte verifica. Vediamo un esempio:
- si assegna una parte con spessore = 20.0 mm
- l'utilizzo della parte in un cluster applica lo spessore del cluster
- l'utilizzo della parte in piazzamenti diretti applica lo spessore della parte.

La determinazione dei piazzamenti residui delle parti utilizzate in cluster è effettuata prima di valutare le reali possibilità di piazzamento dei cluster medesimi. Se per un cluster non sono utilizzati tutti i piazzamenti richiesti, i piazzamenti residui delle singole parti non cambiano.

2.12 Nesting incrementale

Tpa_N permette di avviare una ottimizzazione con piazzamenti già assegnati.
I metodi disponibili sono due ed utilizzabili anche combinati tra loro:
- inserire direttamente una lista di piazzamenti su una o più lastre (piazzamenti preliminari)
- inserire le parti già piazzate come aree di vincolo su una o più lastre

Da un punto di vista esterno, le caratteristiche dei due metodi sono molto differenti ed è evidente come corrispondano ad esigenze e situazioni anche molto differenti.
In Tpa_N, lo schema di piazzamenti di partenza è generabile in modo non definito:
- con inizio manuale di un nesting (su una o più lastre) e successiva richiesta di ultimare la procedura in automatico
- da un nesting generato dalla libreria e poi modificato a mano in una o più lastre e successiva richiesta di ultimare la procedura sulle lastre modificate
- nesting generato da altro software e da integrare con aggiunta di parti e/o cluster e/o lastre
- situazione mista tra le precedenti.

I piazzamenti preliminari

È possibile assegnare una lista di piazzamenti preliminari di parti e/o cluster manuali, tra quelli assegnati nel progetto di nesting.

Un piazzamento preliminare:
- è assegnato tramite appositi metodi e strutture
- è applicato ad una lastra specifica, tra quelle assegnate per il progetto di nesting
- utilizza una parte o un cluster assegnato per il progetto di nesting, decrementandone la quantità residua disponibile
- assegna il piazzamento tramite trasformate geometriche del tutto equivalenti a quelle di un piazzamento calcolato dalla procedura di nesting (traslazione, speculare, rotazione).

La verifica di validità dei piazzamenti preliminari è effettuata in avvio di ottimizzazione (vedi: Criteri e priorità di ottimizzazione).

Vediamo un riassunto sia delle verifiche che sono eseguite che delle caratteristiche che la funzionalità offre:
- è possibile assegnare piazzamenti preliminari su una o più lastre
- le lastre con piazzamenti preliminari devono essere assegnate con quantità disponibile unitaria e sono utilizzate come prime
- un piazzamento può interessare una parte o un cluster: non sono utilizzabili piazzamenti extra e le trasformate richieste devono corrispondere alle assegnazioni dell'elemento (possibilità di ruotare, speculare)
- un piazzamento deve verificare la corrispondenza con la lastra in cui sono assegnati (vedi: materiale, colore, spessore, vincolo di venatura, …)
- con ottimizzazione True Shape è possibile utilizzare le isole della parte (se non occupate da altri piazzamenti preliminari)
- sono comunque eliminati piazzamenti che risultano totalmente esterni all'area utile della lastra corrispondente
- non è possibile chiedere una valutazione preventiva sulla effettiva utilizzabilità di un piazzamento preliminare.

Il posizionamento di un piazzamento preliminare avviene tramite struttura NestPlaced:
- le quote (X, Y) sono applicate assolute sulla lastra, riferite al vertice LB (inferiore sinistro) del rettangolo di ingombro della lastra
- speculare e rotazione sono interpretati con gli stessi criteri dei piazzamenti calcolati dalla libreria.

È responsabilità dell'applicazione chiamante garantire che i piazzamenti preliminari su una lastra non creino sovrapposizioni indesiderate.

L'assegnazione di piazzamenti preliminari può risultare abbastanza incompatibile in abbinamento ad un raggruppamento di parti, ma è comunque il client che, nel caso, deve filtrare o segnalare situazioni di questo tipo. Se si verifica la coesistenza delle due funzionalità, Tpa_N
- assegna i piazzamenti preliminari di una lastra al primo raggruppamento gestito per la lastra
- è prassi che alcune procedure di ottimizzazione di calcolo siano limitate o escluse, con conseguente riduzione dell'efficienza del nesting.

I piazzamenti preliminari compaiono nei risultati del nesting.

Aree di vincolo su una lastra

I piazzamenti già contenuti su una lastra sono ora inseriti come aree di sfrido della lastra medesima, con proprie caratteristiche tecnologiche di ingombro esterno e dimensione dell'utensile.

Ogni piazzamento è assegnato come geometria di sfrido (vedi metodo:IniGeometry) e Tpa_N eviterà di effettuare piazzamenti in queste aree.

I piazzamenti qui assegnati sono in aggiunta alla lista di parti del progetto e non compaiono nei risultati del nesting.

2.13 L'ottimizzazione in Tpa_N

L'ottimizzazione in Tpa_N ha lo scopo primario di ottenere il migliore utilizzo complessivo del materiale disponibile (*lastre*) con i piazzamenti richiesti (*parti*) e con le condizioni poste (*settaggi generali*).

L'ottimizzazione esegue essenzialmente un numero di *tentativi* e sceglie il risultato del tentativo che offre il miglior utilizzo e, quindi, la *soluzione migliore*.
La sequenza dei vari tentativi modifica alcuni *parametri* in modo da generare situazioni di piazzamenti differenti.
La natura di questi che definiamo generalmente *parametri* è varia e può riguardare aspetti anche molto complessi. Alcuni esempi di *parametri* di più immediata comprensione sono:
- modifiche dell'ordine di piazzamento delle parti
- modifiche della logica di riempimento di un foglio
- modifiche sull'angolo di rotazione di una parte
- modifiche sulla direzione del Nesting (orizzontale o verticale).

Un aspetto di fondamentale importanza è la modalità con cui queste variazioni dei parametri sono determinate:
- deterministica
- casuale.

Una modifica eseguita con modalità deterministica è riproducibile invariata nel tempo.
Una modifica eseguita con modalità casuale è riproducibile nel tempo solo in termini di probabilità.

Al momento, entrambe le ottimizzazioni applicano solo modifiche deterministiche.

Per l'ottimizzazione *True Shape* sono allo studio modifiche che possono implementare modifiche anche casuali.

Ottimizzazione Square

La procedura di nesting *Square* è tale da portare alla definizione di una soluzione determinabile in modo ripetitivo, mantenendo invariate tutte le impostazioni al contorno che possono condizionarne lo sviluppo.

I tentativi attuati per questo tipo di ottimizzazione sono studiati in modo da sondare i miglioramenti possibili.

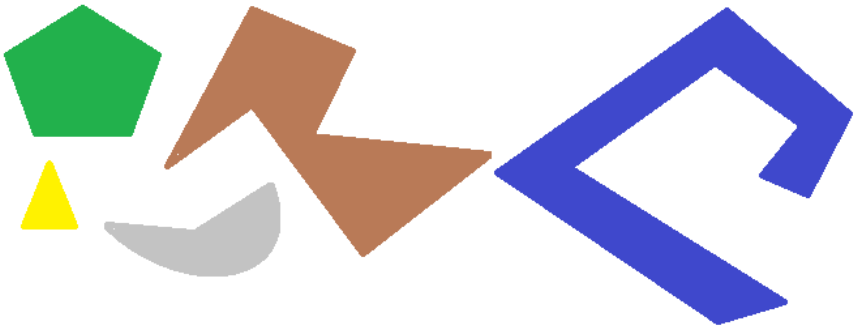
Il numero dei tentativi è comunque contenuto.

In caso di ottimizzazione Square è possibile scegliere tra due differenti modalità di funzionamento, con utilizzo del settaggio RetrySquare:
- *false*: il tempo impostato a determinazione della soluzione non limita il numero dei tentativi predefiniti
- *true*: è possibile che una richiesta di ottimizzazione non esaurisca tutti i tentativi possibili, in conseguenza all'applicazione di un limite temporale.

Nel secondo caso, ulteriori soluzioni possono essere calcolate con tentativi successivi al primo.
Se non altrimenti specificato, nel documento si ipotizza valida la situazione corrispondente a: *RetrySquare* =true.

Ottimizzazione True Shape

La soluzione di nesting è ora resa molto più complessa dal fatto che il piazzamento riguarda *forme libere, concave o convesse, con buchi interni sfruttabili, in numero e possibilità di rotazioni variabili*. Le possibilità di incastro tra le forme sono pressoché infinite. La figura propone 5 differenti forme



È evidente come il problema di piazzare una singola forma per tipo, con rotazione possibile a step di 90° e occupando l'area minore e più compatta possibile, non sia per nulla semplice: ogni forma ha 4 possibili soluzioni di piazzamento (0°, 90°, 180°, 270°) e ognuna deve essere valutata per ogni possibile piazzamento delle restanti figure e per ogni possibile combinazione di accostamento.
Assegniamo ora un numero di ripetizioni per ogni forma (poniamo 10): ogni ripetizione di una forma deve essere valutata con ogni altra ripetizione di forma piazzabile, per un numero complessivo di 50 forme.
Assegniamo ora uno step angolare di 45°: ogni forma ha ora 8 possibili soluzioni di piazzamento (0°, 45°, 90°, …, 315°).
È evidente che il problema si è complicato di molto. In linea teorica è universalmente riconosciuto che la soluzione a un problema di questo tipo non può essere affidata al solo ragionamento, ma sia necessario affidarsi anche al *caso*.

Una prima conseguenza nell'adottare un simile metodo è che non può esserci garanzia che una soluzione sia proposta in modo ripetitivo. Una ulteriore conseguenza è che c'è sempre la possibilità che con un nuovo tentativo si possa migliorare una soluzione: ecco perché è necessario porre un termine di tempo e perché si prevede la possibilità di aggiungere ulteriore tempo alla determinazione della soluzione, partendo dall'ultima determinata, in modo da rendere possibile la valutazione tra soluzioni differenti.

Come già detto: valutazioni di tipo anche casuale sono al momento escluse. I tentativi posti in atti per la determinazione di soluzioni alternative applicano modifiche ripetibili e, di conseguenza, soluzioni ripetibili. Il numero delle soluzioni che si possono calcolare varia sulla base di molte considerazioni, ma al momento è determinato in numero finito. Rimane il fatto che la questione temporale è ora di fondamentale importanza: portare a termine tutti i tentativi necessari può richiedere parecchio tempo ed è quindi necessario porre un termine di tempo.

Ottimizzazioni progressive

Con ottimizzazione *True Shape* è possibile calcolare più soluzioni, fino a un massimo di 20.
Il numero di soluzioni ottenibili con ottimizzazione *Square* è sempre inferiore a 20.
Per la prima ottimizzazione è possibile selezionare tra due funzionamenti (vedi funzione: Compute)
- restituire la soluzione migliore calcolata nel tempo disponibile
- restituire la prima soluzione calcolata, attivando una funzionalità *StepByStep*

Anche per ottimizzazioni successive è possibile selezionare tra differenti funzionamenti (vedi funzione: RetryCompute)
- a tempo o passo-passo
  - a tempo: l'ottimizzazione si arresta sulla base del tempo massimo assegnato
  - passo-passo: restituire la prima soluzione che viene calcolata, attivando una funzionalità *StepByStep*.
- restituire una soluzione solo se migliore della precedente.

Ottimizzazioni successive possono essere calcolate se:
- risulta eseguita una prima ottimizzazione
- rimangono invariate tutte le impostazioni (settaggi, liste di parti, cluster e fogli)
- risultano calcolate meno di 20 ottimizzazioni
- TPA_N non ne ha disattivato in autonomia la possibilità, sulla base di proprie valutazioni di fattibilità ed effettiva utilità.

Ogni ottimizzazione progressiva:
- parte dallo stato lasciato dall'ultima ottimizzazione eseguita
- per quanto detto, non è garantito che sia considerabile *migliore* della precedente.

Tutte le ottimizzazioni calcolate rimangono disponibili fino a:
- una nuova ottimizzazione totale (vedi funzione: Compute)
- una richiesta di cancellazione delle soluzioni trovate (vedi funzione: ClearSolution).

Migliore soluzione

Già si è detto che *la soluzione* di una ottimizzazione è il risultato di una scelta operata tra le soluzioni di tutti i tentativi che sono eseguiti durante la fase di ottimizzazione. Il numero e le caratterisitiche dei tentativi effettuati varia in base a vari fattori.

In ottimizzazione *Square*:
- il numero dei tentativi è definito dalla procedura
- il tempo assegnato per l'ottimizzazione è generalmente sufficiente all'esecuzione dei tentativi previsti
- in alternativa è possibile richiedere ottimizzazioni progressive, come di seguito indicato per il caso di ottimizzazione *True Shape*

In ottimizzazione *True Shape*:
- su una prima ottimizzazione (funzione: *Compute*) e anche se non è richiesta la funzionalità step-by-step, in caso di procedura valutata impegnativa viene comunque messo in atto un solo tentativo, in modo da restituire una soluzione nel minor tempo possibile
- su ottimizzazioni progressive (funzione: *RetryCompute*), il modo operativo è selezionato dalla stessa funzione:
  - *a tempo*: sono attivati più tentativi, fino al tempo massimo disponibile. Quando il tempo è esaurito, la procedura lavora per restituire la soluzione migliore tra quelle calcolate nei tentativi che sono arrivati a termine: è ovvio che c'è un tempo minimo per la chiusura dell'ottimizzazione, dato dal tempo necessario a terminare un tentativo;
  - *step-by-step*: è attivato un solo tentativo.

Ogni ottimizzazione restituisce la soluzione ritenuta migliore nell'ambito dell'ottimizzazione stessa. Ciò significa che ottimizzazioni successive possono portare a soluzioni non necessariamente migliori di quelle già calcolate.

In esecuzione di funzione *RetryCompute* è comunque possibile scegliere tra due modalità:

- se l'ottimizzazione è eseguita correttamente, la soluzione calcolata diventa quella corrente
- la soluzione calcolata diventa corrente solo se è valutata *migliore* della precedente

Ma cosa qualifica una situazione migliore di un'altra?

Diciamo subito che la risposta non è sempre ovvia e che i termini di valutazione richiesti possono in parte differire tra le due modalità di ottimizzazione.
Vediamo alcuni criteri molto generali. I punti riportati di seguito sono applicati secondo l'ordine riportato, passando al successivo nel caso in cui non sia stato possibile operare una scelta prevalente sul punto precedente:

- è privilegiata la maggiore area dei piazzamenti. Una soluzione che dispone pezzi a occupazione del 93.0% delle lastre è migliore di una che determina un riempimento all'88.00%
- è privilegiata la soluzione che favorisce la disposizione secondo la direzione selezionata: lungo l'asse Y in caso di direzione Orizzontale, lungo l'asse X in caso di direzione Verticale
- è privilegiata la soluzione "più ordinata" (la valutazione si basa sul confronto degli sfridi entro il rettangolo di ingombro dei pezzi piazzati)
- sono applicati ulteriori criteri di valutazione relativi al reticolo di disposizione dei pezzi, oltre che al numero e dimensione dei pezzi piazzati e agli sfridi interni.

Ogni punto sopra elencato è applicato con un peso relativo variabile e con alcuni margini di tolleranza, in parte fissi e in parte adattati allo specifico progetto di nesting, in modo da permettere la valutazione combinata del maggior numero di soluzioni.
Un esempio molto pratico: il confronto tra le aree dei piazzamenti non è assoluto (92.7 < 93.0) ma applica un'area di tolleranza valutata in base alla dimensione minima dei pezzi da piazzare, oltre che al diametro della fresa di taglio.

Criteri e priorità di ottimizzazione

Su richiesta di ottimizzazione, Tpa_N avvia un'analisi delle liste assegnate e la successiva fase di ottimizzazione.
L'analisi delle liste può riscontrare situazioni di errore, con conseguente annullamento dell'ottimizzazione.

Esaminiamo questa prima fase di analisi distinguendo tra parti, cluster e fogli.

Verifiche sulla lista delle *parti*:
- sono escluse dall'ottimizzazione parti non abilitate (campo *Enable* in struttura *NestPart*)
- sono escluse dall'ottimizzazione parti che non hanno alcuna corrispondenza con *fogli* validi e che non sono utilizzate in cluster (es.: parte con *Fiber*=1 e nessun foglio con stessa impostazione)
- sono escluse dall'ottimizzazione parti con una o entrambe le dimensioni del rettangolo di ingombro inferiore al valore di risoluzione minima (settaggio generale MinResolution)

Verifiche sulla lista dei *cluster*:
- sono esclusi dall'ottimizzazione cluster non abilitati (campo *Enable* in struttura *NestCluster*) o con quantità richiesta nulla (campo *N* in struttura *NestCluster* con valore 0)
- sono esclusi dall'ottimizzazione cluster che non hanno alcuna corrispondenza con *fogli* validi (es.: cluster con *Fiber*=1 e nessun foglio con stessa impostazione)
- sono esclusi dall'ottimizzazione cluster assegnati con un numero insufficiente di parti (inferiore a 2) o che utilizzano parti non identificate, escluse o non disponibili in quantità minima richiesta
- alcune verifiche specifiche sulle parti determina l'esclusione del cluster:
  - un cluster con venatura (Es.: X) non può utilizzare parte con venatura diversa (Es.: Y)
  - un cluster senza venatura non può utilizzare parte con venatura (X o Y).

In realtà, alcune situazioni indicate non portano necessariamente alla semplice esclusione del cluster ma ad una situazione di errore che impedisce alla procedura di proseguire. È possibile recuperare le situazioni di errore in automatico, portando all'esclusione del cluster, assegnando il settaggio generale FixComputeError=*true*.

Verifiche sulla lista dei *fogli*:
- sono esclusi dall'ottimizzazione fogli non abilitati (campo *Enable* in struttura *NestSheet*)
- sono esclusi dall'ottimizzazione fogli con una o entrambe le dimensioni del rettangolo di ingombro inferiore al valore di risoluzione minima * 50.0 (settaggio generale MinResolution)
- sono esclusi dall'ottimizzazione fogli che non hanno alcuna corrispondenza con *parti* e/o *cluster* validi (es.: foglio con *Fiber*=1 e nessuna parte o cluster con stessa impostazione).

Il risultato delle analisi riportate deve potere utilizzare almeno un elemento per ogni lista di parti e lastre:
- in caso di elemento singolo, può essere richiesta una quantità richiesta/disponibile anche nulla (campo *N* in strutture *NestPart, NestSheet*), con conseguente riconoscimento di situazioni di parte e/o foglio singolo
- altrimenti, sono esclusi dall'ottimizzazione gli elementi con quantità richiesta/disponibile nulla.

Verifiche sulla lista dei *piazzamenti preliminari*:
- sono esclusi dall'ottimizzazione piazzamenti che non hanno una corrispondenza valida con la parte (es.: piazzamento ruotato e parte non ruotabile)
- sono esclusi dall'ottimizzazione piazzamenti che non hanno una corrispondenza valida con la lastra (lastra non assegnata oppure non abilitata oppure disponibile in quantità diversa da 1 oppure di materiale/ colore/ venatura non compatibile).

Anche nel caso dei piazzamenti preliminari, come già per i cluster, le situazioni indicate non portano necessariamente alla semplice esclusione del piazzamento, ma ad una situazione di errore che impedisce alla procedura di proseguire. È possibile recuperare le situazioni di errore in automatico, portando all'esclusione del/i piazzamenti interessati, assegnando il settaggio generale FixComputeError=*true*.

L'avvio della procedura di nesting applica criteri specifici per assegnare l'ordine di utilizzo di parti, cluster e fogli, con alcune distinzioni tra le due modalità di ottimizzazione.

Utilizzo delle lastre

L'utilizzo dei fogli rispetta un ordine che può tenere in considerazione differenti situazioni:
- ordina prima i fogli per i quali sono assegnati piazzamenti preliminari
- ordina prima i fogli marcati come sfrido (campo *IsScrap* in struttura *NestSheet*)
- ordina per priorità decrescente/crescente (se *UseOrderSheet=true*)
- ordina per tipo crescente (campo *ID* in struttura *NestSheet*) a parità delle altre condizioni ai punti precedenti.

Il settaggio generale UseBeforeScrap abilita l'applicazione del campo di qualifica dei *fogli* come sfrido.
Il settaggio generale UseOrderSheet abilita l'applicazione della priorità dei *fogli*.
Il settaggio generale ModeOrderItem assegna la regola per l'interpretazione della priorità:
- *false*: ordina per priorità decrescente (prima le lastre con priorità maggiore)
- *true*: ordina per priorità crescente (prima le lastre con priorità minore).

Utilizzo delle parti (Square)

Il termine di parte accomuna qui sia le parti piazzabili singolarmente che i cluster.

L'utilizzo delle parti rispetta un ordine che può tenere in considerazione differenti situazioni:
- il settaggio generale UseRangePart abilita l'applicazione dei raggruppamenti delle *parti*
- il settaggio generale UseOrderPart abilita l'applicazione della priorità delle *parti*
- il settaggio generale ModeOrderItem assegna la regola per l'interpretazione della priorità (vedi sopra)
- il settaggio *AutomaticGrid* abilita al piazzamento a matrice delle *parti*.

Vediamo ora i principali criteri che sono applicati all'ordinamento iniziale delle parti. I punti sono applicati secondo l'ordine riportato, passando al successivo nel caso in cui non è stato possibile effettuare una scelta prevalente sul punto precedente:

- ordinamento per raggruppamento crescente (se *UseRangePart>0*)
- ordinamento per priorità decrescente/ crescente (se *UseOrderPart=true*)
- parti con richiesta di piazzamento a matrice (se *AutomaticGrid=true*)
- i cluster sono piazzati prima delle parti che li compongono
- ordina per area decrescente (prima i pezzi grandi). L'area è valutata anche in corrispondenza con la direzione del nesting:
  - se direzione orizzontale: ordina per altezza decrescente
  - se direzione verticale: ordina per lunghezza decrescente
- parte senza possibilità di rotazione
- parte *quadrata*
- parte con perimetro maggiore
- parte con quantità richiesta maggiore
- parte con tipo crescente (campo *ID* in struttura *NestPart*): gli ID dei cluster sono accodati in ordine crescente a quelli delle parti.

Parti con sola quantità extra sono in coda all'ordinamento.

Le valutazioni che riguardano il confronto tra aree, dimensioni, perimetri sono in realtà più articolate di quanto una lista possa esporre: qui interessa fornire un quadro generale.

Utilizzo delle parti (True Shape)

Il termine di parte accomuna qui sia le parti piazzabili singolarmente che i cluster.
L'utilizzo delle parti rispetta un ordine che tiene in considerazione una serie di differenti situazioni in numero maggiore rispetto al caso *Square*.

Anche ora:
- il settaggio generale UseRangePart abilita l'applicazione dei raggruppamenti delle *parti*
- il settaggio generale UseOrderSheet abilita l'applicazione della priorità delle *parti*
- il settaggio generale ModeOrderItem assegna la regola per l'interpretazione della priorità
- il settaggio AutomaticGrid abilita il piazzamento a matrice delle *parti*.

Prima di esaminare le situazioni possibili, si deve precisare che l'ordine assegnato alle parti è ora da considerarsi solo come situazione iniziale e modificabile con il progredire dei tentativi di ottimizzazione.

Vediamo ora i principali criteri che sono applicati all'ordinamento iniziale delle parti. I punti riportati sono applicati secondo l'ordine di massima qui riportato, passando al successivo nel caso in cui non sia stato possibile effettuare una scelta prevalente sul punto precedente:

- ordinamento per raggruppamento crescente (se *UseRangePart >0*)
- ordinamento per priorità decrescente/crescente (se *UseOrderPart=true*)
- parti con richiesta di piazzamento a matrice (se *AutomaticGrid=true*)
- cluster piazzati prima delle parti che utilizzano
- parte con forma concava o con isole, con possibilità di inclusione nella zona della concavità o nelle isole
- parte con area maggiore
- valutazioni di forme note (rettangoli, poligoni, cerchi, …)
- parte con minore possibilità di rotazione
- parte con quantità richiesta maggiore
- parte con tipo crescente (campo *ID* in struttura *NestPart*): gli ID dei cluster sono accodati in ordine crescente a quelli delle parti.

Parti con sola quantità extra sono in coda all'ordinamento.

2.14 Supporto per la gestione dei progetti di nesting

Un progetto di nesting è inteso come l'insieme di tutte le informazioni assegnate alla libreria Tpa_N:
- settaggi di abilitazioni complessive
- liste di parti, fogli, abbinamenti manuali (cluster).

Tpa_N espone metodi di serializzazione di un progetto di nesting. Questi metodi sono di particolare utilità per:
- situazioni di testing
- casi di integrazione di Tpa_N in cui le informazioni qui previste nelle strutture di assegnazione delle liste sono sufficienti.

Si ritiene comunque che l'usuale integrazione di Tpa_N richieda una personalizzazione in tal senso.

Vediamo come operano i metodi di serializzazione di un progetto di nesting in Tpa_N, in particolare per i settaggi di abilitazioni complessive.
Come condizione di default, tutti i settaggi sono associati in modo diretto ad un progetto di nesting:
- con eccezione a quelli sopra raggruppati in paragrafo di *Funzionalità generali*
- il salvataggio di un progetto su file ed il suo caricamento includono tutti i settaggi.

Questo tipo di funzionamento è molto utile per le situazioni di testing: è necessario testare una specifica ottimizzazione ed il file di progetto deve essere reso disponibile completo di tutti i settaggi. Per questo motivo, è opportuno che una applicazione esterna della libreria Tpa_N preveda una simile funzionalità.

Tpa_N espone anche metodi di serializzazione dei soli settaggi generali.

2.15 Limiti conosciuti della libreria

Non c'è dubbio che per casistiche limitate di parti, in numero, dimensioni e forma, un nesting manuale potrebbe ottenere risultati anche migliori di Tpa_N: tuttavia, la situazione si ribalta facilmente aumentando anche di poco la complessità della casistica posta. Non costituisce prerogativa di Tpa_N neppure la capacità di ricomporre un puzzle.
Questo paragrafo contiene l'elenco delle limitazioni note di Tpa_N.

Assegnazione delle parti
- È possibile assegnare fino a un massimo di 500 differenti *parti*
- per ogni *parte*, è possibile assegnare una quantità massima piazzabile pari a 999

- per ogni *parte*, è possibile assegnare una quantità massima di 100 isole
- il profilo di descrizione di una parte o isola di parte può assegnare un massimo di 10000 elementi
- il totale massimo dei piazzamenti richiesti per tutte le *parti* e *cluster* assegnati è pari a 99999.

Assegnazione dei cluster

- è possibile assegnare fino ad un massimo di 500 differenti *cluster*
- per ogni *cluster*, è possibile assegnare una quantità massima piazzabile pari a 999
- il totale massimo dei piazzamenti richiesti per tutte le *parti* e *cluster* assegnati è pari a 99999
- un *cluster* può assegnare un minimo di 2 parti e fino a un massimo di 100
- l'utilizzo di un cluster non è opzionale (cioè: non è possibile scegliere di utilizzare un cluster o le parti singolarmente

Assegnazione dei fogli

- è possibile assegnare fino a un massimo di 100 differenti *fogli*
- per ogni *foglio*, è possibile assegnare una quantità massima utilizzabile pari a 999
- per ogni *foglio*, è possibile assegnare una quantità massima di 100 aree di sfrido
- il profilo di descrizione di un foglio o area di sfrido può assegnare un massimo di 10000 elementi

Assegnazione dei piazzamenti preliminari

- è possibile assegnare fino a un massimo di 999 piazzamenti preliminari

Ottimizzazione Square

- il piazzamento di *parte* con venatura assegnata non applica il settaggio *RctMinimize* attivo, se c'è la possibilità di essere piazzato su foglio con venatura assegnata
- risolve solo parzialmente i *Piazzamenti a matrice*.

Ottimizzazione True Shape

- applica in ridondanza i margini esterni assegnati alle *parti*.

Efficienza di Ottimizzazione

Sono molti gli aspetti che concorrono a ridurre l'efficienza teorica di una ottimizzazione. Vediamo i principali:

- aggregazione di parti in cluster manuali: il piazzamento delle parti singole potrebbe aumentare l'efficienza
- raggruppamento di parti: la penalizzazione non deriva solo dalla separazione forzata in gruppi, ma anche da limitazioni o annullamento di pratiche di ottimizzazione automatica (es.: ottimizzazioni con cambio di direzione)
- assegnazione di margini esterni alle parti
- limitazioni imposte da abbinamento di venatura
- limitazione all'utilizzo di zone concave delle parti
- assegnazione di piazzamenti extra: sebbene rispondenti a motivazioni specifiche, i piazzamenti corrispondenti a quantità extra portano ad una riduzione di efficienza, rispetto al piazzamento normale delle stesse quantità.

3 Guida alle funzioni della libreria

Il capitolo descrive in dettaglio proprietà e funzioni di Tpa_N.

3.1 Gestione della licenza

IsSquareEnabled

Proprietà di tipo ***boolean*** che testa presenza e stato della chiave per funzionalità di base (ottimizzazione *Square*).

Valore
True se il modulo è abilitato, *False* altrimenti.

Note
L'interrogazione della proprietà esegue una effettiva lettura della chiave, ma solo se non è in corso una ottimizzazione di nesting.
Nessuna ottimizzazione può essere eseguita se la chiave per funzionalità base non è valida.

IsShapeEnabled

Proprietà di tipo ***boolean*** che testa presenza e stato della chiave per funzionalità avanzata (ottimizzazione *True Shape*).

Valore
True se il modulo è abilitato, *False* altrimenti.

Note
L'interrogazione della proprietà esegue una effettiva lettura della chiave, ma solo se non è in corso una ottimizzazione di nesting.
Alcune azioni della libreria non sono eseguite se la chiave per funzionalità avanzata non è valida.

3.2 Costanti

Le costanti seguenti sono disponibili nella classe TpaNestingOEM.Nesting:

MAX_ROW_ITEMS	500	Numero massimo di tipi di parti assegnabili in progetto di nesting
MAX_ROW_N	2000	Valore massimo di piazzamenti richiesti per una parte
MAX_ROW_ITEMSxN	99999	Valore massimo di piazzamenti totali richiesti (parti e cluster)
MAX_SHEET_ITEMS	100	Numero massimo di tipi di fogli assegnabili in progetto di nesting
MAX_SHEET_N	999	Quantità massima disponibile per un foglio
MAX_CLUSTER_ITEMS	500	Numero massimo di tipi di cluster assegnabili in progetto di nesting
MAX_CLUSTER_N	2000	Valore massimo di piazzamenti richiesti per un cluster
MAX_ROW_INCLUSTER	100	Numero massimo di parti aggiunte in assegnazione di un cluster
MAX_PLACED_N	999	Numero massimo di piazzamenti preliminari assegnabili in progetto di nesting
MAX_ITEMS_INGEO	10000	Numero massimo di elementi geometrici di un profilo
MAX_HOLE_N	100	Numero massimo di isole di parte o area di sfrido di foglio
SOLUTION_NUMBER	20	Numero massimo di soluzioni calcolabili

3.3 Enumerazioni

Le enumerazioni seguenti sono disponibili nel namespace ***TpaNestingOEM***.

NestErrors

Assegnazione degli errori gestiti dalla libreria

- *ErrorNone*: nessun errore
- *ErrorLicense*: licenza base non verificata
- *ErrorLicenseHight*: licenza avanzata non verificata
- *ErrorReset*: procedura di ottimizzazione interrotta dall'utente
- *ErrorMemory*: errore di memoria

- *ErrorPartsEmpty*: lista pezzi vuota o nessun pezzo abilitato o piazzabile
- *ErrorPartsTomany*: troppi piazzamenti di parti richiesti
- *ErrorSheetsEmpty*: lista lastre vuota o nessuna lastra abilitata
- *ErrorSheetsTomany*: troppi fogli richiesti
- *ErrorSheetsMatch*: nessun match di corrispondenza con la lista delle lastre
- *ErrorInGeometry*: errore in assegnazione di elemento geometrico (troppe isole, troppi elementi, linea nulla, arco non valido, …)

- *ErrorClustersTomany*: troppi piazzamenti di cluster richiesti
- *ErrorInCluster*: errore in assegnazione di cluster manuale (parte o quantità minima non disponibile)
- *ErrorClusterMatch*: errore in assegnazione di cluster manuale, nella verifica di corrispondenze specifiche (cluster con venatura assegnata differente da venatura di parti)

- *ErrorPlacedTomany*: troppi piazzamenti preliminari richiesti
- *ErrorPlacedMatch*: errore in verifica di piazzamento preliminare (disponibilità di lastra e/o parte, corrispondenza tra lastra e parte, …)

- *ErrorIOProject*: errore in gestione di file di progetto (errore di accesso a percorso e/o file, formato non valido)
- *ErrorIOfile*: errore in gestione dei file temporanei (errore di accesso a percorso e/o file, …) o accessori (file: DXF)
- *ErrorNoneSolution*: nessuna soluzione risulta assegnata
- *ErrorBusy*: il componente è occupato in ottimizzazione
- *ErrorContext*: segnala che il contesto corrente non è valido
- *ErrorUnexpected*: errore generale o non gestito.

3.4 Strutture

Le strutture seguenti sono disponibili nel namespace ***TpaNestingOEM***.

NestPart

Assegnazione generale di una *parte*. La struttura espone metodi di inizializzazione di campi ai valori di default.

- *int ID*: identificativo numerico della *parte* (univoco, strettamente positivo) {*default* = 0}
  - per il campo è anche utilizzata la dicitura di *tipologia della parte*

- *bool Enable*: abilitazione all'utilizzo della *parte* (false = non piazza la parte) {*default* = true}

- *string Label*: nome descrittivo della parte (può essere =""; lunghezza massima 50 caratteri) {*default* =""}

- *double L*: lunghezza (>= 0.0) {*default* = 0.0}

- *double H*: altezza (>= 0.0) {*default* = 0.0}

- è possibile lasciare i campi (L, H) a 0.0 se per la *parte* è assegnata una geometrica poligonale (es: cerchio, poligono, policurva)
- i valori sono in unità di: Unit

- *double S*: spessore (>= 0.0) {*default* = 0.0}.
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo dei *fogli*
  - il valore è in unità di: Unit

- *int N*: quantità richiesta per piazzamenti (>= 0) {*default* = 0; valore massimo = 999}

- *int Nmax*: quantità massima disponibile {*default* = 0; valore massimo = 999} (si veda settaggio generale *ModeExtraPart*)
  - il campo *N* imposta la quantità richiesta per la parte. Il campo *Nmax* imposta la quantità massima utilizzabile ed è significativo se assegna un valore strettamente positivo. In questo caso:

    - se è *ModeExtraPart* =*false*: *Nmax* deve assegnare un valore maggiore di N e (*Nmax* - *N*)= quantità utilizzabile a riempimento dei *fogli* assegnati.

    - se è *ModeExtraPart* =*true*: *Nmax*= quantità utilizzabile a riempimento dei *fogli* assegnati

  solo dopo avere cercato di piazzare le quantità richieste di tutte le tipologie di *parti*. I piazzamenti corrispondenti all'applicazione del campo *Nmax* sono anche indicati come *piazzamenti extra*

- *short Fiber*: materiale (>= 0) {*default* = 0}
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo dei *fogli*
  - il reale significato attribuito al campo compete all'applicazione esterna

- *int Colour*: generica informazione di colore (>= -1) {*default* = -1}
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo dei *fogli*. Il campo assegna valore di default = -1 per permettere l'assegnazione di valore intero realmente corrispondente a un colore (in questo caso: valore 0 può corrispondere al colore *Nero*).
  - il reale significato attribuito al campo compete all'applicazione esterna

- *short Grain*: direzione venatura (0 = none; 1 = orizzontale; 2 = verticale) {*default* = 0}
  - assegnare il campo con valore 1 o 2 se è necessario applicare una corrispondenza con il corrispondente campo dei *fogli*. Una parte con campo (1, 2) è piazzabile su *foglio* con venatura solo se è possibile rispettare la direzione della venatura stessa (con eventuale applicazione con rotazione); una parte senza venatura è applicabile a lastre con campo *Grain* qualsiasi

- *short Order*: priorità di utilizzo (>= 0) {*default* = 0}
  - *parti* con priorità maggiore/minore hanno precedenza di piazzamento nella soluzione del nesting
  - l'applicazione del campo è condizionata dal settaggio ***UseOrderPart***
  - la regola per l'interpretazione del campo è assegnata dal settaggio ***ModeOrderItem***

- *short Rotate*: rotazione (0 = none; 1 = 90°; 2 = any) {*unità* = grado e decimali di grado; *default* = 0}
  - il valore 2 corrisponde ad abilitare la possibilità di rotazione a passi di un angolo notevole (assegnato con proprietà ***StepAngle***) ed è utilizzato solo in caso di nesting *True Shape*. In caso di nesting *Square*, l'interpretazione è come per il valore 1
  - il valore 1 corrisponde ad abilitare la rotazione di 90°: in caso di nesting *True Shape*, corrisponde alla possibilità di rotazione a passi di 90°

- *short Mirror*: piazzamento speculare (0 = none; 1 = x, 2 = y, 3 = x+y) {*default* = 0}
  - il campo assegna una eventuale trasformata da applicare alla parte originale.

- *bool UseIsle*: abilitazione ad effettuare piazzamenti nelle isole della parte {*default* = false}
  - il campo è significativo solo in caso di nesting *True Shape* e solo se per la *parte* sono assegnate isole
  - l'applicazione del campo è condizionata dal settaggio ***PartInHole***

- *bool AutoPair*: abilitazione all'applicazione della *parte* con modalità di *abbinamento automatico* {*default* = false}
  - il campo è significativo solo in caso di nesting *True Shape*
  - l'applicazione del campo è condizionata dal settaggio ***AutomaticCluster***

- *bool AutoGrid*: abilitazione all'applicazione della *parte* secondo uno sviluppo a matrice {*default* = false}
  - l'applicazione del campo è condizionata dal settaggio ***AutomaticGrid***

- *double PartDiameter*: diametro di taglio specifico della parte {*default* = 0.0}

- *double IsleDiameter*: diametro di taglio specifico delle isole della parte {*default* = 0.0}

- *short Range*: valore di raggruppamento tra elementi (parti e/o cluster) (>= 0) {*default* = 0}
  - l'applicazione del campo è condizionata dal settaggio ***UseRangePart***
  - il reale significato attribuito al campo compete all'applicazione esterna

- *double EdgeL*, *EdgeR*, *EdgeB*, *EdgeT*: ingombri esterni alla parte {*default* = 0.0}.

NestSheet

Assegnazione generale di un *foglio*. La struttura espone metodi di inizializzazione di campi ai valori di default.

- *int ID*: identificativo numerico del *foglio* (univoco, strettamente positivo) {*default* = 0}
  - per il campo è anche utilizzata la dicitura di *tipologia del foglio*

- *bool Enable*: abilitazione all'utilizzo del *foglio* (false = non piazza il foglio) {*default* = true}

- *string Label*: nome descrittivo del *foglio* (può essere =""; lunghezza massima 50 caratteri) {*default* =""}

- *bool IsScrap*: identifica il foglio come recuperato (Es.: sfrido di lastra ottimizzata in precedenza) {*default* = false}
  - *fogli* di questo tipo possono avere precedenza di utilizzo nella soluzione del nesting
  - l'applicazione del campo è assegnata dal settaggio ***UseBeforeScrap***

- *double L*: lunghezza (>= 0.0) {*default* = 0.0}

- *double H*: altezza (>= 0.0) {*default* = 0.0}
  - è possibile lasciare i campi (L, H) a 0.0 se per il foglio è assegnata una geometria poligonale (es: cerchio, poligono, policurva)
  - il valore sono in unità di: Unit

- *double S*: spessore (>= 0.0) {*default* = 0.0}.
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo delle *parti*
  - il valore è in unità di: Unit

- *int N*: quantità disponibile (>= 0) {*default* = 0; valore massimo = 100}

- *short Fiber*: materiale (>= 0) {*default* = 0}
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo delle *parti*
  - il reale significato attribuito al campo compete all'applicazione esterna

- *int Colour*: generica informazione di colore (>= -1) {*default* = -1}
  - assegnare il campo con valori differenziati se è necessario applicare una corrispondenza con il corrispondente campo delle *parti*. Il campo assegna valore di default = -1 per permettere l'assegnazione di valore intero realmente corrispondente a un colore (in questo caso: valore 0 può corrispondere al colore *Nero*).
  - il reale significato attribuito al campo compete all'applicazione esterna

- *short Grain*: direzione di venatura (0 = none; 1 = x = orizzontale; 2 = y = verticale) {*default* = 0}
  - assegnare il campo con valore 1 o 2 se è necessario applicare una corrispondenza con il corrispondente campo delle *parti*

- *short Order*: priorità di utilizzo (>= 0) {*default* = 0}
  - *fogli* con priorità maggiore/minore hanno precedenza di utilizzo nella soluzione del nesting

- l'applicazione del campo è condizionata dal settaggio ***UseOrderSheet***
- la regola per l'interpretazione del campo è assegnata dal settaggio ***ModeOrderItem***

- *double Border*: margine esterno (>= 0.0) {*default* = 0.0}
  - assegnare il campo (con valore strettamente positivo), se è necessario utilizzare un margine esterno differenziato per il foglio
  - il valore è in unità di: Unit

- *double IsleBorder*: margine applicato alle isole del foglio (>= 0.0) {*default* = 0.0}
  - il valore è in unità di: Unit

- *double IsleDiameter*: tecnologia da utilizzare per le isole del foglio (>= 0.0) {*default* = 0.0}
  - il valore è in unità di: Unit

NestCluster

Assegnazione generale di un *abbinamento manuale* (*cluster*): per dettagli sui singoli campi, si veda NestPart.

La struttura espone metodi di inizializzazione di campi ai valori di default.

- *int ID*: identificativo numerico del cluster (univoco, strettamente positivo) {*default* = 0}
  - per il campo è anche utilizzata la dicitura di *tipologia del cluster*

- *bool Enable*: abilitazione all'utilizzo del *cluster* (false = non utilizza) {*default* = true}

- *string Label*: nome descrittivo del *cluster* (può essere =""; lunghezza massima 50 caratteri) {*default* =””}

- *double S*: spessore (>= 0.0) {*default* = 0.0}.

- *int N*: quantità richiesta per piazzamenti (>= 0) {*default* = 0; valore massimo = 999}

- *int Nmax*: quantità massima disponibile {*default* = 0; valore massimo = 999}

- *short Fiber*: materiale (>= 0) {*default* = 0}

- *int Colour*: generica informazione di colore (>= -1) {*default* = -1}

- *short Grain*: direzione venatura (0 = none; 1 = orizzontale; 2 = verticale) {*default* = 0}

- *short Order*: priorità di utilizzo (>= 0) {*default* = 0}

- *short Rotate*: rotazione (0 = none; 1 = 90°; 2 = any) {*unità* = grado e decimali di grado; *default* = 0}

- *short Mirror*: piazzamento speculare (0 = none; 1 = x, 2 = y, 3 = x+y) {*default* = 0}

- *bool UseIsle*: abilitazione ad effettuare piazzamenti nelle isole del cluster {*default* = false}

- *bool AutoPair*: abilitazione all'applicazione del *cluster* con modalità di *abbinamento automatico* {*default* = false}

- *bool AutoGrid*: abilitazione all'applicazione del *cluster* secondo uno sviluppo a matrice {*default* = false}
  - si veda: campi analoghi in struttura *NestPart*

- *short Range*: valore di raggruppamento tra elementi (parti e/o cluster) (>= 0) {*default* = 0}

ItemCluster

Assegnazione di singolo elemento in *cluster manuale*. La struttura espone metodi di inizializzazione di campi ai valori di default.

- *string NameID*: identificativo della parte

  - numerico (strettamente positivo): corrisponde al campo *ID* in struttura *NestPart*
  - altrimenti: corrisponde al campo *Label* in struttura *NestPart*

- *double X*, *Y*: quote (X, Y) di piazzamento del vertice LB (inferiore sinistro) del rettangolo di ingombro della parte
  - le quote sono relative al sistema XY di coordinate cartesiane

- *short Mirror*: speculare richiesto per la parte
  - 0 = none; 1 = x; 2 = y; 3 = x+y
  - (X, Y) fissa l'asse per applicare la trasformata
  - la trasformata è applicata prima della rotazione

- *double A*: angolo di rotazione richiesto per la parte (unità: gradi e decimali di grado)
  - il segno assegna la rotazione: positivo = antioraria; negativo = oraria
  - (X, Y) è il centro della rotazione.

NestSize

Assegnazione del rettangolo di ingombro di una *parte*

- *double LX, LY*: quote (X, Y) di ingombro minimo
- *double HX, HY*: quote (X, Y) di ingombro massimo

NestGeometry

Assegnazione di singolo elemento geometrico in geometria poligonale

- *bool Isle*: true = la geometria corrisponde a un'isola
- *int Type*: tipologia dell'elemento della geometria
  - *0* = elemento iniziale della geometria ((X;Y) = punto di inizio del poligono)
  - *1* = elemento lineare
  - *2* = arco con rotazione oraria
  - *3* = arco con rotazione antioraria
- *double X, Y*: coordinate (X, Y) finale dell'elemento
- *double Xc, Yc*: coordinata (X, Y) del centro dell'elemento (se arco)

NestBox

Assegnazione generale di singolo piazzamento su un *foglio*. La struttura espone metodi di inizializzazione di campi ai valori di default

- *int ID*: identificativo numerico della *parte* corrispondente al piazzamento

- *int Item*: progressivo del piazzamento (>0)

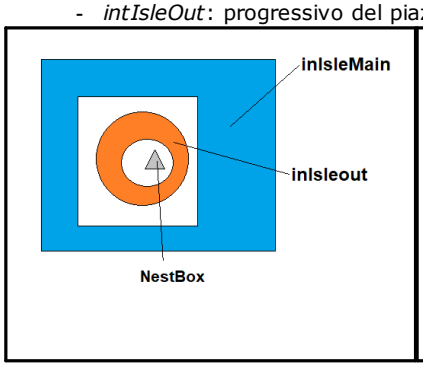
- *double X, Y*: quote (X, Y) di piazzamento del vertice LB (inferiore sinistro) del rettangolo di ingombro originale della parte

- *short Mirror*: speculare richiesto per il piazzamento
  - 0 = none; 1 = x; 2 = y; 3 = x+y
  - l'asse per applicare la trasformata è passante per il centro del rettangolo di ingombro della parte
    - l'asse è verticale per speculare x
    - l'asse è orizzontale per speculare y
  - la trasformata è applicata prima della rotazione

- *double A*: angolo di rotazione corrispondente al piazzamento (unità: gradi e decimali di grado)
  - il segno assegna la rotazione: positivo = antioraria; negativo = oraria
  - (X, Y) è il centro della rotazione

- *bool InIsle*: true = il piazzamento è in un'isola (solo in caso di nesting *True Shape*). I due campi successivi permettono di ricostruire il livello di utilizzo delle isole

- *int InIsleMain*: progressivo del piazzamento più esterno
- *intIsleOut*: progressivo del piazzamento immediatamente esterno (può essere uguale a *InIsleMain*)



In figura un esempio di piazzamento ricorsivo in isole

- *NestBox*: indica la parte piazzata più interna (triangolo grigio)
- *inIsleOut*: è campo in struttura *NestBox* e corrisponde al progressivo della parte immediatamente esterna (cerchio arancione con isola circolare)
- *inIsleMain*: è campo in struttura *NestBox* e corrisponde al progressivo della parte più esterna (rettangolo blu con isola rettangolare)

- *bool IsOver* : true= il piazzamento corrisponde a un extra

- *bool IsInput*: true= corrisponde a un piazzamento preliminare

- *string InCluster*: specifica se il piazzamento deriva dall'applicazione di un cluster
  - "": il piazzamento è singolo
  - altrimenti: deriva dall'esplosione di un cluster. Formato: "#;ID"
    - #: progressivo di applicazione di cluster (> 0). Piazzamenti con stesso valore derivano dallo stesso piazzamento di cluster
    - ID: identificativo numerico del cluster (campo *ID* di struttura *NestCluster*).

- *doble Xl, Yl*: quote (X, Y) di minimo ingombro del piazzamento

- *double Xu, Yu*: quote (X, Y) di massimo ingombro del piazzamento.

NestPlaced

Assegnazione generale di singolo piazzamento preliminare su un *foglio*. La struttura espone metodi di inizializzazione di campi ai valori di default

- *bool Type*: tipologia dell'elemento: false= parte, true=cluster

- *int ID*: identificativo numerico dell'elemento corrispondente al piazzamento (parte o cluster)

- *bool Enable*: abilitazione all'utilizzo del piazzamento (false = non utilizza) {*default* = true}

- *double X, Y*: quote (X, Y) di piazzamento del vertice LB (inferiore sinistro) del rettangolo di ingombro originale della parte

- *short Mirror*: speculare richiesto per il piazzamento

- *double A*: angolo di rotazione corrispondente al piazzamento (unità: gradi e decimali di grado)
  - per dettagli sul significato dei campi, si veda NestBox.

3.5 Funzioni di CallBack

La gestione delle funzioni di CallBack disponibili non è necessaria per il funzionamento di TPA_N.

Progres

Funzione che permette di gestire la progressione dell'ottimizzatore.

void Progres (int nValue, ref bool bCancel, ref bool bPause)

Argomenti

- *nValue*: tempo di ottimizzazione
- *bCancel*: restituire *true* per annullare la procedura
- *bPause*: restituire *true* per interrompere la procedura

Note
La gestione dell'evento permette di annullare o interrompere la procedura di ottimizzazione:
- l'annullamento determina la chiusura della fase di ottimizzazione con cancellazione totale della procedura
- l'interruzione determina la chiusura della fase di ottimizzazione al completamento della prima soluzione valida.

In entrambi i casi è possibile che la chiusura dell'ottimizzazione non sia immediata, essendo comunque attivate modalità di terminazione in sicurezza.
L'intervallo tra occorrenze consecutive dell'evento è regolato dalla proprietà TimerProgres.

3.6 Definizione delle funzionalità

Esaminiamo qui come assegnare tutte le funzionalità di Tpa_N: in riferimento alle funzionalità si utilizza anche la dicitura di settaggi.
In fase di inizializzazione di Tpa_N, tutti i settaggi sono assegnati ai valori di default, di seguito indicati con locuzione {*default* = …}.

Tutte le assegnazioni riportate nel presente capitolo richiedono di essere utilizzate entro una sezione (***IniSettings -> EndSettings***), con eccezione delle proprietà indicate nel raggruppamento *Funzionalità generali* se non altrimenti specificato.
L'utilizzo delle proprietà in lettura non è soggetto a limitazione.

Al fine di fornire un quadro più organico, le assegnazioni sono suddivise in quattro raggruppamenti:

- *Funzionalità generali*: assegnazioni di utilizzo generico
- *Funzionalità relative alle assegnazioni generali di un progetto di nesting*: assegnazioni che hanno significato di personalizzazioni generali di un progetto di nesting
- *Funzionalità relative a nesting Square*: assegnazioni che si applicano ad ottimizzazione di tipo *Square*
- *Funzionalità relative a nesting True Shape*: assegnazioni che si applicano ad ottimizzazione di tipo *True Shape*.

IniSettings

La funzione apre la sezione di assegnazione dei settaggi di funzionamento generale.

bool IniSettings (int Unit, bool Clear, bool Autoconv)

Argomenti
- *Unit*: unità lineare (0 = mm; 1 = inch) {*default* = 0}
- *Clear*: true = assegna i settaggi ai valori di default
- *AutoConv*: true = esegue la conversione automatica dei settaggi dimensionali

Valore di ritorno
True se l'esito è positivo, *False* altrimenti

Note
Un ritorno *True* della funzione permette di proseguire con l'assegnazione totale o parziale dei settaggi di funzionamento generale.

- se risulta attiva una fase di assegnazione di *parti*, *cluster* o *fogli*: la stessa viene preventivamente terminata
- se è *AutoConv=true*: la funzione esegue la conversione automatica dei settaggi dimensionali, ma solo se è *Clear=false* e se *Unit* cambia l'unità rispetto a quella in presa. Nulla è fatto in automatico per le parti e i fogli.

Per l'argomento *Unit* è valutato il valore 0 (zero): Unit=0 assegna unità [mm]; altrimenti assegna unità [inch].
Un cambio dell'unità lineare richiede una assegnazione totale di:
- settaggi dimensionali (se non è: *AutoConv=true*)
- liste di *parti*, *cluster* e *fogli*.

Se è *Clear=true*: tutti i settaggi sono preventivamente assegnati ai valori di default, con eccezione di alcuni settaggi che rimangono invariati: FixComputeError, TimerProgres, DirectoryTemp, RetrySquare.

La fase di assegnazione è chiusa con chiamata della funzione ***EndSettings()***.

Un ritorno *False* della funzione corrisponde a una delle situazioni di errore:
- è in corso una ottimizzazione di nesting.

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

EndSettings

La funzione chiude la sezione di assegnazione dei settaggi di funzionamento generale.

bool EndSettings ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti (la chiamata non ha corrispondenza con ***IniSettings***).

Funzionalità generali

Version

Proprietà di tipo ***string*** che restituisce la versione della libreria.
La stringa riporta il numero principale, secondario, di build e di revisione della libreria.

LicenseType

Proprietà di tipo ***boolean*** che verifica il tipo di licenza che è gestita dalla libreria.
- false: la libreria gestisce la licenza di tipo hardware (chiave USB)
- true: la libreria gestisce la licenza di tipo software (versioni di libreria: 4.0.0, 4.1.0).

La proprietà è disponibile per la verifica della correttezza del contesto in cui si opera.

ExpirationDateString

Proprietà di tipo ***string*** che restituisce la data di scadenza della licenza. La data è formattata usando le convenzioni delle impostazioni di cultura del computer.
Il valore della proprietà è significativo (cioè: diverso da " "), se è riconosciuta una licenza a tempo determinato.
Valore corrispondente a stringa vuota può corrispondere a licenza non verificata o rilasciata senza limitazione temporale.

IsSquareEnabled

Proprietà di tipo ***boolean*** che verifica la validità della licenza di base (ottimizzazione *Square*).

IsShapeEnabled

Proprietà di tipo ***boolean*** che verifica la validità della licenza avanzata (ottimizzazione *True Shape*).

LastError

Proprietà di tipo ***NestErrors*** che restituisce l'ultimo errore riscontrato. Il valore è aggiornato in tutte le procedure che possono diagnosticare una situazione anomala.

ErrorMessage

La funzione restituisce il messaggio assegnato per l'errore indicato.

string ErrorMessage (NestErrors Error)

Argomenti
- *Error*: valore di enumerazione

Valore di ritorno
Se è *Error = NestError.ErrorNone*, il ritorno della funzione è "" (stringa vuota).
I messaggi interni sono assegnati in lingua inglese.

FixComputeError

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a risolvere in automatico situazioni di errore riscontrate in fase di verifica delle liste assegnate.
Il settaggio è al momento utilizzato per risolvere verifiche in lista dei cluster manuali e dei piazzamenti preliminari ed evitare segnalazione di errori: *NestErrors.ErrorInCluster*, *NestErrors.ErrorClusterMatch, NestErrors.ErrorPlacedMatch.*

TimeNesting

Proprietà di tipo ***intero*** che assegna/restituisce il tempo massimo di calcolo di una ottimizzazione di nesting {*unità* = secondi; *default* = 30; *range di validità*: 0 - 600}.
L'assegnazione non è effettuata se è in corso una ottimizzazione di nesting.

Si distinguono i casi particolari:
- valore <= 0 oppure = 600: utilizza il valore massimo (600)
- valore in intervallo (1-20): utilizza il valore minimo (20).

TimerProgres

Proprietà di tipo ***intero*** che assegna/restituisce l'intervallo di tempo per la generazione di eventi ***Progres*** {*unità* = secondi; *default* = 10; *range di validità*: >= 5}.
L'assegnazione non è effettuata se è in corso una ottimizzazione di nesting.

In caso di valore <= 0: disattiva la gestione dell'evento ***Progres***.

RetrySquare

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione alla modalità di ottimizzazione Square {*default* = False}
- *false*: il tempo impostato a determinazione della soluzione non limita il numero dei tentativi predefiniti
- *true*: è possibile che una richiesta di ottimizzazione non esaurisca tutti i tentativi possibili, in conseguenza dell'applicazione di un limite di tempo.

L'assegnazione non è effettuata se è in corso una ottimizzazione di nesting.
Il valore di default corrisponde alla modalità attiva nelle versioni di TPA_N precedenti l'introduzione della proprietà (< ver 4.0.1). Il valore consigliato è: *true.*

DirectoryTemp

Proprietà di tipo ***string*** che assegna/restituisce la cartella per la gestione dei file temporanei {*default* =""}.
Se non è assegnata una cartella valida (assegna stringa vuota oppure la cartella non è accessibile), utilizza il percorso di archiviazione della libreria TPA_N.
L'assegnazione della proprietà richiede l'utilizzo entro una sezione (***IniSettings -> EndSettings***).

Funzionalità relative alle assegnazioni generali di un progetto di nesting

Unit

Proprietà di tipo ***intero*** che restituisce l'unità assegnata con funzione *IniSettings* {*default* = 0}.

MinResolution

Proprietà di tipo ***double*** che assegna/restituisce il valore di risoluzione minima {*unità*: Unit; *default* = 0.1 [mm]; *range di validità*: 1.0E - 5 / 1.0 [mm]}.
Il campo è utilizzato per valutare la validità di un rettangolo (valore minimo di lunghezza/altezza) o di un elemento geometrico (es.: tratto lineare nullo o arco non valido).

OneNestingDimension

Proprietà di tipo ***boolean*** che assegna/restituisce la selezione relativa a due differenti tipologie di foglio {*default* = False}:
- (false) 2D: i fogli assegnano una parte di piano su cui il nesting è bidimensionale
- (true) 1D: il nesting è su supporto lineare ed i fogli assegnano barre, profilati, tubi.

OneCutterDimension

Proprietà di tipo ***boolean*** che assegna/restituisce la tipologia dell'utensile di taglio delle parti {*default* = False}:
- (false) l'utensile ha dimensione in XY (Es.: fresa, laser)
- (true) l'utensile ha una sola dimensione (Es.: lama, taglierino).

CutterDiameter

Proprietà di tipo ***double*** che assegna/restituisce il diametro di taglio delle parti, da utilizzarsi in assenza di assegnazioni valide per le singole parti {*unità*: Unit; *default* = 0.0; *range di validità*: >= 0.0}.
Il piazzamento tra *parti* contigue applica un distanziamento minimo corrispondente al valore impostato.

TecnoDistanceType

Proprietà di tipo ***boolean*** che assegna/restituisce la modalità di applicazione della tecnologia delle parti {*default* = False}
- *False*: l'utensile è applicato attorno al profilo di ogni parte
- *True*: l'utensile è applicato a sormonto del profilo di ogni parte.

OverrunPolygon

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione di sormonto delle aree di taglio delle parti {*default* = False}.
La sovrapposizione massima consentita corrisponde al diametro di taglio, con sottrazione della *Distanza di sicurezza* (assegnata con proprietà ***OverrunSecurity***).
La figura illustra la differenza di comportamento, al variare del valore della proprietà:

- a sinistra i piazzamenti con valore *False*
- a destra i piazzamenti con valore *True*.

OverrunSecurity

Proprietà di tipo ***double*** che assegna/restituisce la *Distanza di sicurezza* aggiunta ai piazzamenti con applicato il sormonto delle aree di tagli delle parti {*unità*: Unit; *default* = 0.1 [mm]; *range di validità*: 0.0/10.0 [mm]}

OverrunSheet

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione per i profili di taglio delle parti a sormontare i margini dei fogli {*default* = True}.
- *True*: i profili di taglio delle parti possono sormontare i margini esterni di un foglio. Il calcolo della distanza minima tra un bordo del foglio ed un piazzamento corrisponde alla metà del diametro di taglio (CutterDiameter * 0.5) o al diametro stesso, in base al settaggio MaximizeOverrunSheet.
- *False*: i profili di taglio delle parti sono applicati entro i margini esterni di un foglio. La distanza minima tra un bordo del foglio e un piazzamento corrisponde al diametro di taglio (CutterDiameter).

Come già detto, la determinazione della distanza minima dei piazzamenti dai bordi può aggiungere un distanziamento in considerazione di vari fattori:
- assegnazione di margini per l'utilizzo delle lastre (vedi: MarginLeft(), …)
- modifica di tratti di profili con applicazione dell'*errore cordale*
- assegnazione di diametri di tecnologia differenti per le parti.

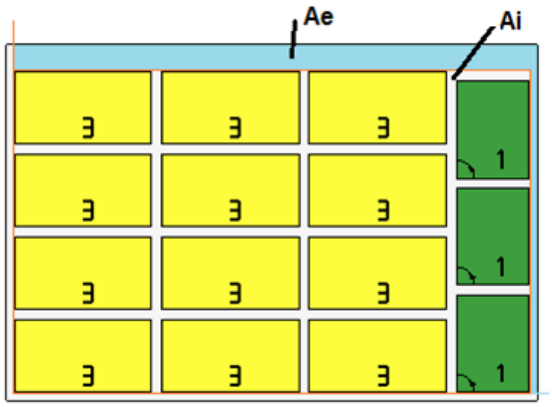
MaximizeOverrunSheet

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a massimizzare il sormonto dei profili di taglio ai margini dei fogli. L'impostazione è significativa se è *OverrunSheet*=*true*.

SolutionMaxScrap

Proprietà di tipo ***double*** che assegna/restituisce il valore per la Massima differenza di scarti interni {*unità*: %; *default* = 2.5; *range di validità*: 0.5/50.0}.
La valutazione è effettuata calcolando gli sfridi in percentuale del rettangolo di ingombro che include i piazzamenti di un foglio. L'utilizzo dell'informazione è abbinato agli altri criteri che sono presi in considerazione per determinare la scelta migliore tra soluzioni differenti.
L'utilizzo dell'informazione in questi termini può essere scarsamente efficace in caso di foglio non rettangolare, in quanto il rettangolo di ingombro dei piazzamenti può includere anche aree non utili al piazzamento, in quanto esterne al profilo del foglio.

Con riferimento alla figura:



- il rettangolo esterno rappresenta il foglio
- **Ai** è l'area interessata ai piazzamenti: è delimitata dalle coordinate limite delle parti. La differenza tra l'area **Ai** e l'area di tutti i piazzamenti corrisponde all'area degli *sfridi interni* al nesting
- **Ae** è l'area esterna ai piazzamenti e corrisponde all'area degli *sfridi esterni* al nesting

SolutionExpected

Proprietà di tipo ***double*** che assegna/restituisce il valore minimo atteso di utilizzo dei fogli {*unità*: %; *default* = 75.0; *range di validità*: 25.0/95.0}.
La valutazione è effettuata calcolando l'area dei piazzamenti in percentuale del rettangolo di ingombro che include tutti i piazzamenti di un foglio (**Ai** del disegno precedente). L'impostazione è utilizzata per valutazioni relative dello stato di riempimento di un foglio o come condizione aggiunta di chiusura della fase di calcolo, in alternativa al raggiungimento del tempo massimo di calcolo.

ExtraFiller

Proprietà di tipo ***boolean*** che assegna/restituisce la modalità di applicazione dei piazzamenti extra {*default* = False}.
- *False*: i piazzamenti extra sono applicati a riempimento dei fogli
- *True*: i piazzamenti extra sono applicati solo a riempimento della lunghezza o altezza già impegnata con i pezzi richiesti. La direzione di riempimento è scelta in base alla direzione di avanzamento per i piazzamenti (vedi: proprietà Direction):
  - se Orizzontale: il riempimento è applicato alla lunghezza del foglio, mentre non è applicato limite in altezza
  - se Verticale: il riempimento è applicato all'altezza del foglio, mentre non è applicato limite in lunghezza.

ExtraFillerLastSheet

Proprietà di tipo ***boolean*** che assegna/restituisce la modalità di applicazione del settaggio *ExtraFiller* {*default* = False}.
- *False*: applica *ExtraFiller* a tutte le lastre
- *True*: applica *ExtraFiller* solo all'ultima lastra (se non ripetuta).

ModeExtraPart

Proprietà di tipo ***boolean*** che assegna/ restituisce il criterio di interpretazione dei piazzamenti extra per le parti ed i clusters {*default*=False}:
- *False*: l'impostazione corrisponde al totale massimo complessivo dei piazzamenti effettuabili (inclusi i piazzamenti richiesti)
- *True*: l'impostazione corrisponde al massimo dei piazzamenti effettuabili, da sommare ai piazzamenti richiesti.

UseOnlyExtraPart

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione ad utilizzare parti con soli piazzamenti extra assegnati {*default*=False}:
- *false*: l'applicazione di piazzamenti extra può solo sommarsi a piazzamenti richiesti
- *true*: l'applicazione di piazzamenti extra non richiede necessariamente piazzamenti richiesti

L'impostazione non riguarda i cluster manuali, per i quali è da intendersi applicata un'interpretazione automatica a *false:* l'utilizzo di un cluster manuale richiede sempre l'impostazione di piazzamenti richiesti.

ModeMirrorPart

Proprietà di tipo ***boolean*** che assegna/restituisce il criterio di applicazione dello speculare delle parti o dei cluster {*default* = False}:
- *False*: l'applicazione della trasformata è richiesta ed applicata
- *True*: l'applicazione della trasformata è conseguente all'impossibilità di effettuare piazzamenti in modalità non speculata.

ModeOrderItem

Proprietà di tipo ***boolean*** che assegna/restituisce il criterio di interpretazione della priorità di parti, cluster, lastre {*default* = False}:
- *false*: ordina per priorità decrescente (prima gli elementi con priorità maggiore)
- *true*: ordina per priorità crescente (prima gli elementi con priorità minore).

UseOrderPart

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione della priorità di *parti* e *cluster* {*default* = True}.
- *true*: abilita l'applicazione del campo *Order* assegnato in strutture (***NestPart***, ***NestCluster***) di assegnazione di una parte o cluster
- *false*: il campo della struttura è ignorato.

UseOrderSheet

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione della priorità dei *fogli* {*default* = True}.
- *true*: abilita l'applicazione del campo *Order* assegnato in struttura ***NestSheet*** di assegnazione di un foglio
- *false*: il campo della struttura è ignorato.

UseBeforeScrap

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione ad utilizzare prima i fogli marcati di sfrido {*default* = False}.
- *true*: abilita l'utilizzo del campo *IsScrap* assegnato in struttura ***NestSheet*** di assegnazione di un foglio
- *false*: il campo della struttura è ignorato.

MatchType

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione di corrispondenza di *materiale* tra parti e fogli {*default* = True}.
- *true*: abilita l'applicazione del campo *Fiber* assegnato in strutture (***NestPart***, ***NestCluster***, ***NestSheet***)
- *false*: i campi delle strutture sono ignorati.

MatchColor

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione di corrispondenza di *colore* tra parti e fogli {*default* = True}.
- *true*: abilita l'applicazione del campo *Colour* assegnato in strutture (***NestPart***, ***NestCluster***, ***NestSheet***)
- *false*: i campi delle strutture sono ignorati.

MatchGrain

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione di corrispondenza di *venatura* tra parti e fogli {*default* = False}.
- *true*: abilita l'applicazione del campo *Grain* assegnato in strutture (***NestPart***, ***NestCluster***, ***NestSheet***)
- *false*: i campi delle strutture sono ignorati.

UseRangePart

Proprietà di tipo ***short*** che assegna/restituisce il criterio di riconoscimento dei raggruppamenti di singole parti e/o cluster {*default* = 0}.
- *0*: la funzionalità è disabilitata (disattiva l'interpretazione del campo *Range)*
- *1:* richiede piazzamenti in aree continue
- *2:* richiede piazzamenti in aree separate da un canale.

DimRangePart

Proprietà di tipo ***double*** che assegna/restituisce la dimensione del canale di separazione tra aree accostate con applicazione di *UseRangePart*= 2 {*default* = 0.0}.

RctMinimize

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione alla ricerca della rotazione che corrisponde all'ingombro minimo delle *parti* {*default* = True}. L'abilitazione è applicabile alle parti con assegnata una geometria primaria non rettangolare e con possibilità di ruotare.

MarginOuter

Proprietà di tipo ***double*** che assegna i margini esterni ai fogli {*unità*: Unit; *default* = 0.0; *range di validità*: >= 0.0}.
Il valore assegna complessivamente i quattro margini esterni ai fogli.

MarginLeft, MarginRight, MarginBottom, MarginTop

Proprietà di tipo ***double*** che assegnano/restituiscono i margini esterni ai fogli, rispettivamente sul lato sinistro, destro, basso e alto {*unità*: Unit; *default* = 0.0; *range di validità*: >= 0.0}.

In caso di ottimizzazione *TrueShape*, l'utilizzo di margini differenziati è applicabile a *fogli* rettangoli. In caso di fogli di forma generica, si applica un valore unico che corrisponde al massimo impostato tra i quattro.

MarginInner

Proprietà di tipo ***double*** che assegna/restituisce una distanza aggiuntiva applicata tra i piazzamenti {*unità*: Unit; *default* = 0.0; *range di validità*: >= 0.0}.

Direction

Proprietà di tipo ***short*** che assegna/restituisce la direzione di avanzamento per i piazzamenti {*default* = 0; *range di validità*: 0/1}:
- 0 = direzione orizzontale
- 1 = direzione verticale.

In caso di settaggio *OneNestingDimension=true*: la direzione è applicata con valore 1 (direzione verticale).

Corner

Proprietà di tipo ***short*** che assegna/restituisce il vertice di partenza per i piazzamenti, tra i valori validi {*default* = 0; *range di validità*: 0/3}
- 0 = Sinistro-Inferiore
- 1 = Sinistro-Superiore
- 2 = Destro-Inferiore
- 3 = Destro-Superiore.

Funzionalità relative a nesting Square

Non ci sono al momento funzionalità previste.

Funzionalità relative a nesting True Shape

StepAngle

Proprietà di tipo ***double*** che assegna/restituisce l'angolo minimo di rotazione {*default* = 5.0; *unità* = grado e decimali di grado; *range di validità*: 1.0/90.0}.
L'impostazione corrisponde al valore 2 del campo *Rotate* assegnato in struttura ***NestPart*** di assegnazione di una parte e in struttura NestCluster di assegnazione cluster.

PartInHole

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a effettuare piazzamenti entro le aree di sfrido delle parti {*default* = True}.
- *true*: abilita l'applicazione del campo *UseIsle=true* assegnato in struttura ***NestPart*** di assegnazione di una parte e in struttura *NestCluster* di assegnazione cluster.

PartInHoleMulti

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a effettuare piazzamenti *ricorsivi* entro le aree di sfrido delle parti {*default* = False}.

PartInHoleBefore

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a *privilegiare* i piazzamenti entro le aree di sfrido delle parti {*default* = False}.

AutomaticCluster

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione degli *Abbinamenti automatici* {*default* = True}.
- *true*: abilita l'applicazione del campo *AutoPair=true* assegnato in struttura ***NestPart*** di assegnazione di una parte ed in struttura NestCluster di assegnazione cluster.

ClustersExpected

Proprietà di tipo ***double*** che assegna/restituisce il valore di utilizzazione minima dell'area (in %) di un abbinamento automatico di parti rispetto ai piazzamenti singoli {*default* = 50.0; *range di validità*: 50.0/95.0}.
L'impostazione assegna l'efficienza minima che è valutata per l'*Abbinamento automatico* di una parte, che è calcolata come:

(Area della singola parte * 2 * 100) / (Lunghezza gruppo * Altezza gruppo)

ClustersAbsolute

Proprietà di tipo ***double*** di significato analogo al precedente (*ClustersExpected*) per l'applicazione in autonomia di cluster automatici {*default* = 50.0; *range di validità*: 50.0/100.0}.

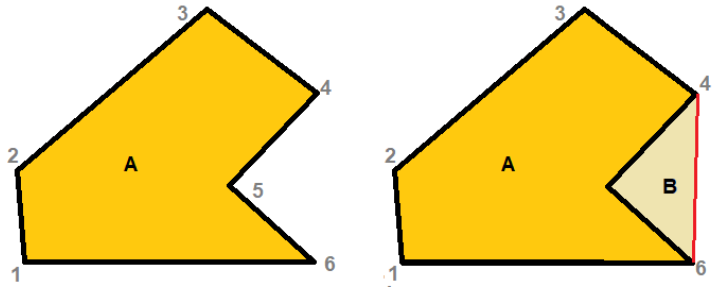
Il valore impostato non dovrebbe essere inferiore al precedente.

AutomaticGrid

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione all'applicazione dei *Piazzamenti a matrice* {*default* = True}.
- *true*: abilita l'applicazione del campo *AutoGrid=true* assegnato in struttura ***NestPart*** di assegnazione di una parte ed in struttura NestCluster di assegnazione cluster
- false: esclude l'applicazione di piazzamenti a matrice programmati.

ExploreConcave

Proprietà di tipo ***boolean*** che assegna/restituisce l'abilitazione a esplorare le concavità di una parte {*default* = True}.
In figura il caso di una parte concava:
- *true*: la parte è utilizzata come assegnata (a sinistra della figura: vertici indicati da 1 a 6)
- *false*: la parte è utilizzata con eliminazione della concavità. Ai fini della procedura di nesting, la parte è modificata come risulta sulla destra della figura: la lista dei vertici elimina il punto 5 e l'ingombro totale della parte aggiunge l'area indicata con la lettera B.



L'esclusione all'utilizzo delle zone di concavità velocizza le procedure di calcolo, a discapito di una minore efficienza di utilizzo del materiale.

ConcaveDimension

Proprietà di tipo ***double*** che assegna/restituisce una lunghezza utilizzata per la riduzione delle concavità di una parte {*unità*: Unit; *default* = 1.0; *range di validità*: >= 0.0}.
Il valore è utilizzato in caso di ***ExploreConcave****=true*: la procedura di nesting esplora le aree concave ma opera una semplificazione delle stesse.
Con riferimento alla concavità riportata nella figura precedente, l'eliminazione della concavità è condizionata a verifiche:

- l'area della parte B è piccola rispetto all'area del cerchio di raggio pari al valore impostato; oppure
- la distanza tra il vertice 5 e il tratto che unisce i vertici (4, 6) è inferiore al valore impostato.

Funzioni di serializzazione dei settaggi

SaveSettings

La funzione salva tutti i settaggi in file (formato XML)

NestErrors SaveSettings (string pathName, bool bMode)

Argomenti
- *pathName*: percorso del file
- *bMode*: *** argomento disponibile

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione di assegnazione (***IniSettings***, ***IniSetPart***, ***IniSetSheet***, ***IniSetCluster***)
- (*NestError. ErrorIOProject*) si è verificato un errore nell'accesso al file da scrivere.

Dal salvataggio sono comunque esclusi alcuni settaggi: FixComputeError, TimerProgres, DirectoryTemp, RetrySquare.

LoadSettings

La funzione legge le assegnazioni di tutti i settaggi da file (formato XML).

NestErrors LoadSettings (string pathName, bool bMode)

Argomenti
- *pathName*: percorso del file
- *bMode*: *** argomento disponibile

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione di assegnazione (***IniSettings***, ***IniSetPart***, ***IniSetSheet***, ***IniSetCluster***)
- (*NestError.ErrorIOProject*) si è verificato un errore nell'accesso al file da leggere.

Tutti i settaggi sono preventivamente assegnati ai valori di default.
Dalla lettura sono comunque esclusi alcuni settaggi, che rimangono invariati: FixComputeError, TimerProgres, DirectoryTemp, RetrySquare.

3.7 Definizione delle parti

IniSetPart

La funzione apre la sezione di assegnazione delle parti.

bool IniSetPart (bool Clear)

Argomenti
- *Clear*: true = azzera la lista delle parti

Valore di ritorno
True se l'esito è positivo, *False* altrimenti

Note

Un ritorno *True* della funzione permette di proseguire con l'assegnazione totale o parziale delle parti.
La fase di assegnazione è chiusa con chiamata della funzione ***EndSetPart()***.

Un ritorno *False* della funzione corrisponde a una delle situazioni di errore:
- si è verificato uno stato grave di sistema con conseguente errore in allocazione di memoria
- è in corso una ottimizzazione di nesting.

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

EndSetPart

La funzione chiude la sezione di assegnazione delle parti.

bool EndSetPart ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti (la chiamata non ha corrispondenza con ***IniSetPart*** oppure la lista delle parti non è valida).

AddPart

La funzione aggiunge una parte alla lista.

NestErrors AddPart (NestPart Item)

Argomenti
- *Item*: struttura di assegnazione della parte

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPart -> EndSetPart***)
- (*NestError.ErrorPartsTomany*) la lista delle parti è già al massimo consentito (500 elementi) o la quantità richiesta in struttura eccede il massimo consentito (999)
- (*NestError.ErrorUnexpected*) per la parte è assegnato un identificativo non valido (***Item.ID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) risulta già assegnata una parte con identificativo numerico ***Item.ID***.

L'assegnazione della parte può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.

RemovePart

La funzione elimina una parte o le geometrie di una parte.

NestErrors RemovePart (int ItemID, bool OnlyGeometry)

Argomenti
- *ItemID*: identificativo della parte (> 0)
- *OnlyGeometry*: true = elimina solo le assegnazioni aggiunte di geometria (se assegnate)

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPart -> EndSetPart***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnata una parte con identificativo numerico ***ItemID***.

WritePart

La funzione modifica una parte già assegnata.

NestErrors WritePart (NestPart Item)

Argomenti
- *Item*: struttura di assegnazione della parte

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno *diverso da NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPart -> EndSetPart***)
- (*NestError.ErrorUnexpected*) non risulta assegnata una parte con identificativo numerico ***Item.ID***.

L'assegnazione della parte può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.

Le assegnazioni aggiunte di geometria rimangono invariate.

CountPart

La proprietà di tipo ***intero*** ottiene il numero delle parti in lista.
L'utilizzo della proprietà non richiede di operare entro una sezione (***IniSetPart -> EndSetPart***).

ReadPart

La funzione cerca una parte in corrispondenza all'ID specificato.

NestErrors ReadPart (*int ItemID, ref *NestPart Item, ref NestSize ItemSize)

Argomenti
- *ItemID*: identificativo della parte (> 0)
- *Item*: struttura di assegnazione della parte
- *ItemSize*: struttura di assegnazione del rettangolo di ingombro utilizzato dalla procedura di ottimizzazione

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetPart -> EndSetPart***).

Un ritorno *diverso da NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnata una parte con identificativo numerico ***ItemID***.

ReadPartIndex

La funzione cerca una parte in corrispondenza all'indice specificato.

NestErrors ReadPartIndex (*int Index, ref *NestPart Item, ref NestSize ItemSize)

Argomenti
- *Index*: indice (base zero) sulla lista delle parti (>= 0)
- *Item*: struttura di assegnazione della parte
- *ItemSize*: struttura di assegnazione del rettangolo di ingombro utilizzato dalla procedura di ottimizzazione

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetPart -> EndSetPart***).
L'utilizzo primario della funzione è per acquisizione delle parti dopo l'esecuzione di funzione ***LoadProject***.

Un ritorno *diverso da NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting

- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un indice non valido.

3.8 Definizione dei fogli

IniSetSheet

La funzione apre la sezione di assegnazione dei fogli.

bool IniSetSheet (bool Clear)

Argomenti
- *Clear*: true = azzera la lista dei fogli

Valore di ritorno
True se l'esito è positivo, *False* altrimenti

Note
Un ritorno *True* della funzione permette di proseguire con l'assegnazione totale o parziale dei fogli.
La fase di assegnazione è chiusa con chiamata della funzione ***EndSetSheet()***.

Un ritorno *False* della funzione corrisponde a una delle situazioni di errore:
- si è verificato uno stato grave di sistema con conseguente errore in allocazione di memoria
- è in corso una ottimizzazione di nesting.

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

EndSetSheet

La funzione chiude la sezione di assegnazione dei fogli.

bool EndSetSheet ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti (la chiamata non ha corrispondenza con ***IniSetSheet*** oppure la lista dei fogli non è valida).

AddSheet

La funzione aggiunge un foglio alla lista.

NestErrors AddSheet (NestSheet Item)

Argomenti
- *Item*: struttura di assegnazione del foglio

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetSheet -> EndSetSheet***)
- (*NestError.ErrorSheetsTomany*) la lista dei fogli è già al massimo consentito (100 elementi) o la quantità richiesta in struttura eccede il massimo consentito (999)
- (*NestError.ErrorUnexpected*) per il foglio è assegnato un identificativo non valido (***Item.ID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) risulta già assegnato un foglio con identificativo numerico ***Item.ID***.

L'assegnazione del foglio può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.

RemoveSheet

La funzione elimina un foglio o le geometrie di un foglio.

NestErrors RemoveSheet (int ItemID, bool OnlyGeometry)

Argomenti
- *ItemID*: identificativo del foglio (> 0)
- *OnlyGeometry*: true = elimina solo le assegnazioni aggiunte di geometria (se assegnate)

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetSheet -> EndSetSheet***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnato un foglio con identificativo numerico ***ItemID***.

WriteSheet

La funzione modifica un foglio già assegnato.

NestErrors WriteSheet (NestSheet Item)

Argomenti
- *Item*: struttura di assegnazione del foglio

Valore di ritorno
True se l'esito è positivo, *False* altrimenti

Note
Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetSheet -> EndSetSheet***)
- (*NestError.ErrorUnexpected*) non risulta assegnato un foglio con identificativo numerico ***Item.ID***.

L'assegnazione del foglio può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.

Le assegnazioni aggiunte di geometria rimangono invariate.

CountSheet

La proprietà di tipo ***intero*** ottiene il numero dei fogli in lista.
L'utilizzo della proprietà non richiede di operare entro una sezione (***IniSetSheet -> EndSetSheet***).

ReadSheet

La funzione cerca un foglio in corrispondenza all'ID specificato.

NestErrors ReadSheet (*int ItemID, ref *NestSheet Item, ref NestSize ItemSize)

Argomenti
- *ItemID*: identificativo del foglio (> 0)
- *Item*: struttura di assegnazione del foglio
- *ItemSize*: struttura di assegnazione del rettangolo di ingombro utilizzato dalla procedura di ottimizzazione

Valore di ritorno
True se l'esito è positivo, *False* altrimenti.

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetSheet -> EndSetSheet***).

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)

- (*NestError.ErrorUnexpected*) non risulta assegnato un foglio con identificativo numerico ***ItemID***.

ReadSheetIndex

La funzione cerca un foglio in corrispondenza all'indice specificato.

NestErrors ReadSheetIndex (*int Index, ref *NestSheet Item, ref NestSize ItemSize)

Argomenti
- *Index*: indice (base zero) sulla lista dei fogli (>= 0)
- *Item*: struttura di assegnazione del foglio
- *ItemSize*: struttura di assegnazione del rettangolo di ingombro utilizzato dalla procedura di ottimizzazione.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetSheet -> EndSetSheet***).
L'utilizzo primario della funzione è per acquisizione dei fogli dopo l'esecuzione di funzione ***LoadProject***.

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un indice non valido.

3.9 Definizione delle geometrie poligonali

Esaminiamo ora come assegnare le geometrie (*forma*) delle parti e/o dei fogli.
Come semplificazione, di seguito verrà assunto di operare in assegnazione della lista delle parti.

IniGeometry

La funzione apre la sezione di assegnazione di una geometria poligonale di una parte.

bool IniGeometry (int ItemID, bool IsIsle, double X, double Y, double IsleBorder, double IsleDiameter)

Argomenti
- *ItemID*: identificativo della parte (> 0)
- *IsIsle*: true = l'assegnazione corrisponde ad isola di parte/area di sfrido di lastra (di geometria primaria già assegnata)
- *X, Y*: coordinate iniziali della geometria (i valori sono in unità di: Unit)
- *IsleBorder*: margine assegnato per la geometria (significativo se: strettamente positivo (i.e.: >0) e *Isle*=true)
- *IsleDiameter*: diametro di taglio (significativo se: strettamente positivo (i.e.: >0) e *IsIsle=true*)

Valore di ritorno
True se l'esito è positivo, *False* altrimenti

Note
Un ritorno *True* della funzione permette di proseguire con l'assegnazione.
La fase di assegnazione è chiusa con chiamata della funzione ***EndGeometry()***.

Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPart -> EndSetPart***, oppure ***IniSetSheet -> EndSetSheet***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnata una parte con identificativo numerico ***ItemID***
- (*NestError.ErrorUnexpected*) se è *IsIsle=false*: non deve risultare già assegnato un profilo primario
- (*NestError.ErrorInGeometry*) se è *IsIsle=true* e la lista delle isole è già al massimo consentito (100 elementi)

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

Se è *IsIsle=true* e non è assegnato un profilo primario, il medesimo viene generato in automatico con forma rettangolare e dimensioni (lunghezza * altezza) assegnate per la parte.

Gli argomenti (*IsleBorder, IsleDiameter*) sono significativi solo in caso di aree interne:
- Se isola di parte: (*IsleDiameter >0)* assegna un diametro di taglio specifico per l'isola; *IsleBorder* non è significativo
- Se sfrido di lastra: (*IsleBorder* >0) assegna un margine di rispetto esterno all'area; (*IsleDiameter >0)* assegna il diametro di taglio per l'area.

Per le casistiche di assegnazione ed utilizzo degli argomenti tecnologici si veda il paragrafo Distanziamenti dalle aree di vincolo interne al foglio.

EndGeometry

La funzione chiude la sezione di assegnazione delle geometrie poligonali di una parte.

bool EndGeometry ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti.

Note
Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- la chiamata non ha corrispondenza con ***IniGeometry***
- la lista delle geometrie assegnate per la parte non è valida.

La funzione esegue un controllo generale di validità delle geometrie complessivamente assegnate per la parte e non solo della singola geometria assegnata con la sessione corrente; per ogni geometria:
- devono risultare assegnati almeno un elemento curvo o due elementi lineari.

AddToGeometry_Line

La funzione aggiunge un elemento lineare a geometria poligonale.

NestErrors AddToGeometry_Line (double Xend, double Yend)

Argomenti
- *Xend*: coordinata X finale del tratto (unità di: Unit)
- *Yend*: coordinata Y finale del tratto (unità di: Unit)

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno diverso da *NestError.ErrorNone* della funzione corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorInGeometry*) la lista degli elementi è già al massimo consentito (10000 elementi)
- (*NestError.ErrorInGeometry*) l'elemento lineare ha lunghezza nulla (<=***MinResolution***).

L'inizio del tratto lineare corrisponde al punto finale del tratto precedente.

AddToGeometry_Arc

La funzione aggiunge un elemento curvo (arco) a geometria poligonale.

NestErrors AddToGeometry_Arc (double Xend, double Yend, double Xcentre, double Ycentre, bool IsCCW)

Argomenti
- *Xend, Yend*: coordinate (X, Y) finale del tratto (unità di: Unit)
- *Xcentre, Ycentre*: coordinata (X, Y) del centro del tratto (unità di: Unit)
- *IsCCW*: true = rotazione antioraria

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorInGeometry*) la lista degli elementi è già al massimo consentito (10000 elementi)
- (*NestError.ErrorInGeometry*) l'elemento curvo non è valido (raggio <= ***MinResolution*** oppure |raggio iniziale - raggio finale| > ***MinResolution***).

L'inizio del tratto curvo corrisponde al punto finale del tratto precedente.

AddToGeometry_Circle

La funzione aggiunge un elemento di cerchio a geometria poligonale.

NestErrors AddToGeometry_Circle (double Xcentre, double Ycentre, bool IsCCW)

Argomenti
- *Xcentre, Ycentre*: coordinata (X, Y) del centro del tratto (unità di: Unit)
- *IsCCW*: true = rotazione antioraria

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Vedi funzione ***AddToGeometry_Arc***.

AddToGeometry

La funzione aggiunge un elemento generico a geometria poligonale.

NestErrors AddToGeometry (NestGeometry Item)

Argomenti
- *item*: struttura di assegnazione dell'elemento geometrico.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Vedi funzioni ***AddToGeometry_****.

Lettura delle geometrie

ReadGeometry

La chiamata ricorsiva della funzione legge le geometrie che risultano assegnate per una parte

NestErrors ReadGeometry (int List, int ItemID, int IndexItem, int IndexElement, ref NestGeometry Item)

Argomenti
- *List*: seleziona la lista: 0 = parti; 1 = fogli
- *ItemID*: identificativo dell'elemento (parte o foglio) (> 0)
- *IndexItem*: indice (base zero) sulla lista dei profili geometrici assegnati
- *IndexElement*: indice (base zero) dell'elemento geometrico in profilo geometrico
- *Item*: struttura di assegnazione dell'elemento geometrico.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo primario della funzione è per acquisizione delle geometrie dopo l'esecuzione di funzione ***LoadProject***.

In corrispondenza a valori assegnati per (*List*, *ItemID*), la prima chiamata deve utilizzare (*IndexItem=0, IndexElement=0*). Un ritorno a questa prima chiamata differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)

- (*NestError.ErrorUnexpected*) identificativo di elemento non valido oppure elemento senza assegnazione di geometria (primaria o secondaria).

ReadGeometryInfo

La chiamata della funzione legge le informazioni accessorie assegnate alla geometria secondaria di una parte

NestErrors ReadGeometryInfo (int List, int ItemID, int IndexItem, ref double IsleBorder, ref double IsleDiameter)

Argomenti
- *List*: seleziona la lista: 0 = parti; 1 = fogli
- *ItemID*: identificativo dell'elemento (parte o foglio) (> 0)
- *IndexItem*: indice (base zero) sulla lista dei profili geometrici assegnati
- *IsleBorder:* margine assegnato per la geometria (vedi: IniGeometry*)*
- *IsleDiameter:* diametro di taglio assegnato alla geometria (vedi: IniGeometry*)*

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo primario della funzione è per acquisizione delle geometrie dopo l'esecuzione di funzione ***LoadProject*** e limitatamente al caso di geometria corrispondente ad isola di parte o area di sfrido di lastra.

Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) identificativo di elemento non valido oppure elemento di assegnazione primaria.

Utilizzo di geometrie esterne

ShapeExtension

Proprietà di tipo ***string*** che restituisce la lista delle estensioni di file supportate per l'interpretazione di geometrie da file esterno. Vediamo i casi al momento possibili:

- *=""*: nessun riconoscimento valido. Una chiamata a funzione *ReadShape* fallisce
- *=".DXF"*: è riconosciuta la lettura di file DXF

ReadShape

La funzione interpreta una geometria da file DXF.

NestErrors ReadShape (string pathOpen)

Argomenti
- *pathOpen*: percorso completo del file da leggere (es: "C:\PATTERN\A.DXF").

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ERR_NONE* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorLicense*) si è verificato esito negativo nella verifica di presenza e stato della chiave per funzionalità avanzata
- (*NestError.ErrorIOfile*) si è verificato un errore nell'accesso o interpretazione del file *pathOpen*
- (*NestError.ErrorIOfile*) si è verificato un errore nella gestione di file temporanei.

La funzione non modifica nessuna assegnazione delle geometrie delle parti: esegue una interpretazione di formato e carica le geometrie lette rendendole disponibili per una successiva lettura.

Vediamo le regole generali di interpretazione di un file di disegno:
- è interpretato un disegno 2D
- sono interpretate solo entità geometriche che assegnano profili (polilinee, cerchi, ellissi, spline)
- sono riconosciuti validi solo profili chiusi (la condizione di profilo chiuso è valutata applicando una distanza massima pari al diametro di taglio: vedi ***CutterDiameter***)
- è mantenuto 1 solo profilo master (quello di area maggiore) e le sue eventuali isole, assegnate al primo livello interno.

ReadGeoInShape

La chiamata ricorsiva della funzione legge le geometrie assegnate con chiamata a funzione *ReadShape*.

bool ReadGeoInShape (int IndexItem, int IndexElement, ref NestGeometry Item)

Argomenti
- *IndexItem*: indice (base zero) sulla lista dei profili geometrici assegnati
- *IndexElement*: indice (base zero) dell'elemento geometrico in profilo geometrico
- *Item*: struttura di assegnazione dell'elemento geometrico.

Valore di ritorno
True se l'esito è positivo, *False* altrimenti.

Note
La prima chiamata deve utilizzare (*IndexItem=0, IndexElement=0*). Un ritorno a questa prima chiamata *false* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorUnexpected*) non è assegnata geometria (primaria o secondaria).

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

3.10 Definizione degli abbinamenti manuali

IniSetCluster

La funzione apre la sezione di assegnazione dei cluster.

bool IniSetCluster (bool Clear)

Argomenti
- *Clear*: true = azzera la lista dei cluster.

Valore di ritorno
True se l'esito è positivo, *False* altrimenti.

Note
Un ritorno *True* della funzione permette di proseguire con l'assegnazione totale o parziale dei cluster.
La fase di assegnazione è chiusa con chiamata della funzione ***EndSetCluster ()***.

Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- si è verificato uno stato grave di sistema con conseguente errore in allocazione di memoria
- è in corso una ottimizzazione di nesting.

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

EndSetCluster

La funzione chiude la sezione di assegnazione dei cluster.

bool EndSetCluster ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti (la chiamata non ha corrispondenza con ***IniSetCluster*** oppure la lista dei cluster non è valida).

AddCluster

La funzione aggiunge un cluster alla lista.

NestErrors AddCluster (NestCluster Item)

Argomenti
- *Item*: struttura di assegnazione del cluster

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno *positivo* della funzione permette di proseguire con l'assegnazione della composizione geometrica del cluster.

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetCluster -> EndSetCluster***)
- (*NestError.ErrorClustersTomany*) la lista dei cluster è già al massimo consentito (500 elementi) o la quantità richiesta in struttura eccede il massimo consentito (999)
- (*NestError.ErrorUnexpected*) per il cluster è assegnato un identificativo non valido (***Item.ID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) risulta già assegnato un cluster con identificativo numerico ***Item.ID***.

L'assegnazione del cluster può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.
Il nuovo cluster non ha assegnata alcuna composizione geometrica.

AddToCluster

La funzione aggiunge un elemento a definizione della composizione geometrica del cluster.

NestErrors AddToCluster (int ItemID, ItemCluster Item)

Argomenti
- *ItemID*: identificativo del cluster (> 0)
- *Item*: struttura di assegnazione della parte.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno diverso da *NestError.ErrorNone* della funzione corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetCluster -> EndSetCluster***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnato un cluster con identificativo numerico ***ItemID***
- (*NestError.ErrorInCluster*) non risulta assegnata una parte con identificativo ***Item.NameID*** (si rammenta che il campo può assegnare un identificativo numerico o alias testuale)
- (*NestError.ErrorInCluster*) la lista degli elementi che assegnano la composizione del cluster è già al massimo consentito (100).

RemoveCluster

La funzione elimina un cluster o gli elementi che ne definiscono la composizione.

NestErrors RemoveCluster (int ItemID, bool OnlyGeometry)

Argomenti
- *ItemID*: identificativo del cluster (> 0)
- *OnlyGeometry*: true = elimina solo le assegnazioni aggiunte di geometria (se assegnate).

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetCluster -> EndSetCluster***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnato un cluster con identificativo numerico ***ItemID***.

WriteCluster

La funzione modifica le assegnazioni di un cluster già assegnato.

NestErrors WriteCluster (NestCluster Item)

Argomenti
- *Item*: struttura di assegnazione del cluster.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetCluster -> EndSetCluster***)
- (*NestError.ErrorUnexpected*) non risulta assegnato un cluster con identificativo numerico ***Item.ID***.

L'assegnazione del cluster può utilizzare valori modificati rispetto alla struttura, in caso di dati non validi.
Le assegnazioni aggiunte di geometria rimangono invariate.

CountCluster

La proprietà di tipo ***intero*** ottiene il numero dei cluster in lista.
L'utilizzo della proprietà non richiede di operare entro una sezione (***IniSetCluster -> EndSetCluster***).

ReadCluster

La funzione cerca un cluster in corrispondenza all'ID specificato.

NestErrors ReadCluster (int ItemID, ref NestCluster Item)

Argomenti
- *ItemID*: identificativo del cluster (> 0)
- *Item*: struttura di assegnazione del cluster.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetCluster -> EndSetCluster***).

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido (***ItemID*** deve essere strettamente positivo)
- (*NestError.ErrorUnexpected*) non risulta assegnato un cluster con identificativo numerico ***ItemID***.

ReadClusterIndex

La funzione cerca un cluster in corrispondenza all'indice specificato.

NestErrors ReadClusterIndex (int Index, ref NestCluster Item)

Argomenti
- *Index*: indice (base zero) sulla lista dei cluster (>= 0)
- *Item*: struttura di assegnazione del cluster.

Valore di ritorno

NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetCluster -> EndSetCluster***).
L'utilizzo primario della funzione è per acquisizione dei cluster dopo l'esecuzione di funzione ***LoadProject***.

Un ritorno diverso da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorUnexpected*) è assegnato un indice non valido.

ReadInCluster

La chiamata ricorsiva della funzione legge gli elementi di composizione che risultano assegnati per un cluster.

NestErrors ReadInCluster (int ItemID, int IndexItem, ref ItemCluster Item)

Argomenti
- *ItemID*: identificativo del cluster (> 0)
- *IndexItem*: indice (base zero) sulla lista degli elementi assegnati in cluster
- *Item*: struttura di assegnazione dell'elemento.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
L'utilizzo primario della funzione è per acquisizione dei clusters dopo l'esecuzione di funzione ***LoadProject***.

In corrispondenza a valore assegnato per (*ItemID*), la prima chiamata deve utilizzare (*IndexItem=0*). Un ritorno a questa prima chiamata differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un identificativo non valido di cluster oppure non sono assegnati elementi di composizione.

3.11 Definizione dei piazzamenti preliminari

IniSetPlaced

La funzione apre la sezione di assegnazione dei piazzamenti preliminari.

bool IniSetPlaced (bool Clear)

Argomenti
- *Clear*: true = azzera la lista dei piazzamenti preliminari.

Valore di ritorno
True se l'esito è positivo, *False* altrimenti.

Note
Un ritorno *True* della funzione permette di proseguire con l'assegnazione totale o parziale dei piazzamenti.
La fase di assegnazione è chiusa con chiamata della funzione ***EndSetPlaced ()***.

Un ritorno *False* della funzione corrisponde ad una delle situazioni di errore:
- si è verificato uno stato grave di sistema con conseguente errore in allocazione di memoria
- è in corso una ottimizzazione di nesting.

Interrogare la proprietà *LastError* per valutare la situazione specifica di errore.

EndSetPlaced

La funzione chiude la sezione di assegnazione dei piazzamenti preliminari.

bool EndSetPlaced ()

Valore di ritorno
True se l'esito è positivo, *False* altrimenti (la chiamata non ha corrispondenza con ***IniSetPlaced*** oppure la lista dei piazzamenti non è valida).

AddPlaced

La funzione aggiunge una parte alla lista.

NestErrors AddPlaced (int ID_sheet, NestPlaced Item)

Argomenti
- *ID_Sheet*: identificativo del foglio (> 0)
- *Item*: struttura di assegnazione del piazzamento

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPlaced -> EndSetPlaced***)
- (*NestError.ErrorPlacedTomany*) la lista è già al massimo consentito (999 elementi)
- (*NestError.ErrorUnexpected*) per il foglio o la parte è assegnato un identificativo non valido (***ID_sheet*** *e* ***Item.ID*** devono essere strettamente positivi).

RemovePlaced

La funzione elimina uno o più piazzamenti preliminari.

NestErrors RemovePlaced (*int Index, *int ID_sheet)

Argomenti
- *Index*: indice (base zero) sulla lista (>= 0)
- *ID_Sheet*: identificativo del foglio (> 0)

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
Un ritorno diverso da *NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorContext*) la funzione non è utilizzata entro una sezione (***IniSetPlaced -> EndSetPlaced***)
- (*NestError.ErrorUnexpected*) sono assegnati argomenti non validi

Si distinguono due situazioni di funzionamento:
- (*Index>=0)* è eliminato il piazzamento corrispondente all'indice indicato in *(Index)*
- (*Index<0, ID_Sheet>0)* sono eliminati tutti i piazzamenti assegnati al foglio indicato in *(ID_Sheet).*

CountPlaced

La proprietà di tipo ***intero*** ottiene il numero dei piazzamenti in lista.
L'utilizzo della proprietà non richiede di operare entro una sezione (***IniSetPlaced -> EndSetPlaced***).

ReadPlacedIndex

La funzione cerca il piazzamento in corrispondenza all'indice specificato.

NestErrors ReadPlacedIndex (*int Index*, ref int ID_sheet, ref NestPlaced Item)

Argomenti
- *Index*: indice (base zero) sulla lista (>= 0)
- *ID_Sheet*: identificativo del foglio (> 0)
- *Item*: struttura di assegnazione del piazzamento

Valore di ritorno
NestError.ErrorNone se l'esito è positivo

Note
L'utilizzo della funzione non richiede di operare entro una sezione (***IniSetPlaced -> EndSetPlaced***).

Un ritorno *diverso da NestError.ErrorNone* corrisponde a una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione (***IniGeometry -> EndGeometry***)
- (*NestError.ErrorUnexpected*) è assegnato un indice non valido.

3.12 Soluzione del nesting

Compute

La funzione avvia la procedura di ottimizzazione

NestErrors Compute (int OptiSelect, bool OnlyTest, bool StepByStep)

Argomenti
- *OptiSelect*: seleziona il tipo di ottimizzazione (-1 = automatico; 0 = Square; 1 = True Shape)
- *OnlyTest*: true = esegue solo assegnazioni preliminari all'ottimizzazione
- *StepByStep*: seleziona il tipo di funzionamento dell'ottimizzazione.

Valore di ritorno
NestError.ERR_NONE se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è già in corso una ottimizzazione di nesting
- (*NestError.ErrorLicense*) si è verificato esito negativo nella verifica di presenza e stato della chiave per funzionalità di base
- (*NestError.ErrorLicenseHight*) è richiesta ottimizzazione *True Shape* e lo stato della chiave non corrisponde a funzionalità avanzata
- (*NestErrors.ErrorPartsEmpty*) la lista delle parti è vuota oppure non assegna elementi abilitati
- (*NestErrors.ErrorSheetsEmpty*) la lista dei fogli è vuota oppure non assegna elementi abilitati
- (*NestErrors.ErrorInCluster*) errore in assegnazione di cluster manuale (parte o quantità minima non disponibile)
- (*NestErrors.ErrorClusterMatch*) errore in assegnazione di cluster manuale, nella verifica di corrispondenze specifiche (cluster con venatura assegnata differente da venatura di parti)
- (*NestErrors.ErrorPlacedMatch*) errore in assegnazione di piazzamenti preliminari, nella verifica di corrispondenze specifiche
- (*NestError.ErrorReset*) la procedura è stata annullata dall'applicazione chiamante.

All'avvio, la funzione
- chiude una eventuale sezione di assegnazioni aperta (settaggi, parti, fogli, cluster)
- effettua le verifiche preliminari (di cui sopra)
- effettua verifiche sulle liste di parti, fogli e cluster
- valuta il tipo di ottimizzazione richiesta, sulla base della verifica della chiave e delle liste di fogli e parti.

Se l'esecuzione delle verifiche elencate porta ad una situazione di errore, comunque la funzione ritorna senza eseguire alcuna ottimizzazione.
In caso contrario, prosegue in base ad ***OnlyTest***:
- ***true***: non modifica lo stato di ottimizzazione corrente e semplicemente ritorna l'esito (positivo o di errore)
- ***false***: annulla lo stato di ottimizzazione eseguita ed avvia una nuova ottimizzazione.

Il valore di ***StepByStep*** seleziona tra due funzionamenti possibili della procedura di ottimizzazione:

- *false*: l'ottimizzazione si arresta sulla base del tempo massimo assegnato

- *true*: l'ottimizzazione si arresta alla prima soluzione che viene calcolata.

La libreria esegue ottimizzazione *Square*:
- se tutte le *parti* e i *fogli* sono assegnati come semplici rettangoli e non sono assegnati cluster ed è *OptiSelect=-1 [NOTA (1)]*
- se è *OptiSelect=0*
- se fallisce la verifica di presenza e stato della chiave per funzionalità avanzata

In esecuzione di ottimizzazione *Square*:
- le *parti* assegnate con un profilo geometrico sono considerate per il rettangolo di ingombro del profilo stesso ed i profili assegnati come isole sono ignorati
- i *fogli* assegnati con un profilo geometrico sono considerati per il rettangolo di ingombro del profilo stesso ed i profili assegnati come aree di vincolo sono ignorati: in questo caso, la soluzione di ottimizzazione può eseguire dei piazzamenti in aree esterne al foglio e/o interni alle aree di vincolo
- i *cluster* sono considerati per il rettangolo di ingombro complessivo della composizione geometrica assegnata
- la possibilità di calcolare ottimizzazioni progressive è determinata dal settaggio RetrySquare, oltre che dalla complessità dell'ottimizzazione.

[NOTA (1)] La valutazione della geometria di parti e fogli e un po' più complessa di quanto sopra detto per eseguire una ottimizzazione Square:
- una parte può assegnare isole, ma è possibile escluderne l'utilizzo per piazzamenti interni (è eseguito un confronto preventivo delle aree delle parti piazzabili e delle isole disponibili)
- il profilo geometrico di una parte è rettangolare o assimilabile ad un rettangolo: rapporto (area profilo/area rettangolo di ingombro) >=85%, progressione dei punti lungo il profilo e scostamento degli stessi dal rettangolo di ingombro entro un sesto della dimensione minima del rettangolo di ingombro
- il profilo geometrico di una lastra è rettangolare o assimilabile ad un rettangolo: tutti tratti lineari, rapporto (area profilo/area rettangolo di ingombro) >=95%, passaggio dai 4 vertici del rettangolo di ingombro, progressione dei punti lungo il profilo e scostamento degli stessi dal rettangolo di ingombro entro la risoluzione impostata.

IsComputed

Proprietà di tipo ***boolean*** che restituisce l'informazione di ottimizzazione eseguita.
Il valore della proprietà è significativo dopo una chiamata a funzione ***Compute*** con ***OnlyTest=false*** ed ottimizzazione eseguita.

ModeCompute

Proprietà di tipo ***intero*** che restituisce l'informazione applicata all'ultima ottimizzazione eseguita: 0 = tipo *Square*; 1 = tipo *True Shape*.
Il valore della proprietà è significativo dopo una chiamata a funzione ***Compute*** con ***OnlyTest=false*** ed ottimizzazione eseguita.

TimeCompute

Proprietà di tipo ***double*** che restituisce il tempo di calcolo corrispondente all'ultima esecuzione di funzione ***Compute*** oppure ***RetryCompute*** {*unità* = secondi}.
Il valore della proprietà è significativo dopo l'esecuzione di una ottimizzazione.

TimeOut

Proprietà di tipo ***boolean*** che restituisce l'informazione se la procedura di calcolo è terminata per raggiunto time-out.
Il valore della proprietà è significativo dopo l'esecuzione di una ottimizzazione.

CanRetry

Proprietà di tipo ***boolean*** che restituisce l'informazione se è possibile richiedere un nuovo tentativo di ottimizzazione, partendo dall'ultima soluzione trovata.
Le condizioni per potere calcolare più soluzioni sono:

- deve risultare eseguita e conclusa una ottimizzazione
- non devono essere state assegnate modifiche al componente (settaggi generali, lista delle parti e/o dei fogli e/o dei cluster)
- non è stato già calcolato il numero massimo gestito di soluzioni (assegnato = 20)
- la procedura di ottimizzazione ha margini operativi per nuovi tentativi di calcolo.

RetryCompute

La funzione avvia la procedura di ottimizzazione, partendo dall'ultima soluzione trovata.

NestErrors RetryCompute (bool StepByStep, bool RetryBest)

Argomenti
- *StepByStep*: seleziona il tipo di funzionamento dell'ottimizzazione
- *RetryBest*: true = assegna nuova soluzione solo se migliore della precedente

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- vedi casi di proprietà ***CanRetry***
- (*NestError.ErrorReset*) la procedura è stata annullata dall'applicazione chiamante.

Il valore di *RetryBest* seleziona tra due funzionamenti possibili:
- *false*: se l'ottimizzazione è eseguita correttamente, la soluzione calcolata diventa quella corrente
- *true*: la soluzione calcolata diventa corrente solo se è valutata *migliore* della precedente.

Il valore di *StepByStep* seleziona tra due funzionamenti possibili della procedura di ottimizzazione:
- *false*: l'ottimizzazione si arresta sulla base del tempo massimo assegnato
- *true*: l'ottimizzazione si arresta alla prima soluzione che viene calcolata.

ClearSolution

La funzione azzera ogni soluzione calcolata.

NestErrors ClearSolution ()

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting.

3.13 I risultati del nesting

Tutte le proprietà e le funzioni qui elencate falliscono in corrispondenza ad una delle situazioni di errore:

- è in corso una ottimizzazione di nesting
- non è calcolata una ottimizzazione valida.

Solution

Proprietà di tipo ***intero*** che assegna/restituisce il numero della soluzione corrente.
Il valore restituito è:
- 0: nessuna soluzione è calcolata
- 1: la soluzione corrente corrisponde alla prima (con chiamata di funzione ***Compute()***)
- >1: la soluzione corrente corrisponde ad una calcolata con chiamata a funzione ***RetryCompute()***.

In assegnazione:
- la proprietà permette di cambiare la *soluzione corrente* ad una già calcolata
- non è effettuata assegnazione se il valore non è valido per l'individuazione di una soluzione.

OfSolution

Proprietà di tipo ***intero*** che restituisce il numero totale delle soluzioni calcolate.
Il valore restituito è:
- 0: nessuna soluzione è calcolata
- 1: è calcolata solo la soluzione con chiamata di funzione ***Compute()***
- >1: risultano calcolate più soluzioni (***Compute()*** seguita da chiamate a funzione ***RetryCompute()***).

Fitness

Proprietà di tipo ***double*** che restituisce l'informazione di efficienza della *soluzione corrente*.
Il valore della proprietà è significativo dopo l'esecuzione di una ottimizzazione.
L'efficienza complessiva è valutata in % del rapporto tra l'area utilizzata per i piazzamenti e l'area totale dei pannelli utilizzati: maggiore è l'area dei piazzamenti in un foglio e maggiore l'efficienza calcolata.

Per l'ultimo foglio di una soluzione, la valutazione di efficienza può opportunamente limitare l'area utile per i piazzamenti al rettangolo di ingombro risultante per i piazzamenti effettuati: diminuendo l'area disponibile, aumenta l'efficienza calcolata per il foglio. Sono però poste ultime limitazioni:

- il foglio è eseguito senza ripetizioni
- il foglio ha forma rettangolare e non ha aree di vincolo.

ReadNumResult

La funzione legge il numero dei fogli della *soluzione corrente* (fogli con piazzamenti).

int ReadNumResult (int ID_Sheet)

Argomenti
- *ID_Sheet*: identificativo del foglio (> 0)

Note
La funzione torna il numero dei fogli della soluzione corrente o di tipo di foglio:

ID_Sheet	
<=0	ottiene il numero complessivo dei fogli calcolati per la soluzione
>0	ottiene il numero dei fogli di tipo (*ID_Sheet*) calcolati per la soluzione

Il valore restituito tiene conto di fogli ripetuti uguali.

ReadNumPartInResult

La funzione legge il numero di piazzamenti in un foglio della soluzione corrente o sull'intera soluzione, con possibilità di corrispondenza a un solo tipo di parte.

int ReadNumPartInResult (int Index, int ID_Part)

Argomenti
- *Index*: indice a base zero del foglio della soluzione
- *ID_Part*: identificativo della parte (> 0)

Note
La funzione legge il numero di piazzamenti corrispondenti agli argomenti assegnati:

Index	*ID_Part*	
-1	0	ottiene il numero totale dei piazzamenti della soluzione
-1	>0	ottiene il numero dei piazzamenti corrispondenti alla parte (*ID_Part*) in tutti i fogli della soluzione

>=0	0	ottiene il numero totale dei piazzamenti in foglio di indice (*Index*) della soluzione
>=0	>0	ottiene il numero totale dei piazzamenti corrispondenti alla parte (*ID_Part*) in foglio di indice (*Index*)

Nei casi di funzionamento con *Index* negativo (-1), il valore restituito tiene conto di fogli ripetuti uguali.
Nei casi di funzionamento con *Index* >= 0, il valore restituito conteggia i piazzamenti di un singolo foglio, senza considerare le eventuali ripetizioni del foglio stesso.

ReadNumClusterInResult

La funzione legge il numero di piazzamenti cluster in un foglio della soluzione corrente o sull'intera soluzione, con possibilità di corrispondenza ad un solo tipo di cluster.

int ReadNumClusterInResult (int Index, int ID_Item)

Argomenti
- *Index*: indice a base zero del foglio della soluzione
- *ID_Item*: identificativo del cluster (> 0).

Note
La funzione legge il numero di piazzamenti di tipo cluster corrispondenti agli argomenti assegnati:

Index	*ID_Item*	
-1	0	ottiene il numero totale dei piazzamenti cluster della soluzione
-1	>0	ottiene il numero dei piazzamenti corrispondenti al cluster (*ID_Item*) in tutti i fogli della soluzione
>=0	0	ottiene il numero totale dei piazzamenti in foglio di indice (*Index*) della soluzione
>=0	>0	ottiene il numero totale dei piazzamenti corrispondenti al cluster (*ID_Item*) in foglio di indice (*Index*)

Nei casi di funzionamento con *Index* negativo (-1), il valore restituito tiene conto di fogli ripetuti uguali.
Nei casi di funzionamento con *Index* >=0, il valore restituito conteggia i piazzamenti di un singolo foglio, senza considerare le eventuali ripetizioni del foglio stesso.

ReadResult

La funzione restituisce le informazioni relative ad un foglio della *soluzione corrente* in corrispondenza dell'indice specificato

bool ReadResult (int Index, ref int ID_Sheet, ref int Item_Sheet, ref double AreaPercent, ref int NumPlaces, ref int Repetition)

Argomenti
- *Index*: indice (base zero) del foglio della soluzione
- *ID_Sheet*: identificativo numerico del *foglio* (campo ID in struttura *NestSheet*)
- *Item_Sheet*: occorrenza del foglio di tipo identificativo numerico del foglio
- *AreaPercent*: area occupata/area totale
- *NumPlaces*: numero di parti piazzate sul foglio
- *Repetition*: numero di ripetizioni del foglio.

Valore di ritorno
True se il foglio è individuato valido.

Note
L'argomento *AreaPercent* è strettamente correlato al calcolo del Fitness: maggiore è l'area dei piazzamenti in un foglio e maggiore è l'efficienza calcolata.

ReadPartInResult

La funzione restituisce le informazioni relative ad un piazzamento su un foglio della *soluzione corrente*

bool ReadPartInResult (int Index, int IndexPart, ref NestBox Item)

Argomenti
- *Index*: indice (base zero) del foglio della soluzione
- *IndexPart*: indice (base zero) del piazzamento sul foglio della soluzione
- *Item*: struttura di assegnazione del piazzamento.

Valore di ritorno
True se il piazzamento sul foglio è individuato valido.

Note
Come già indicato, la struttura *Item* riporta le informazioni relative al piazzamento:

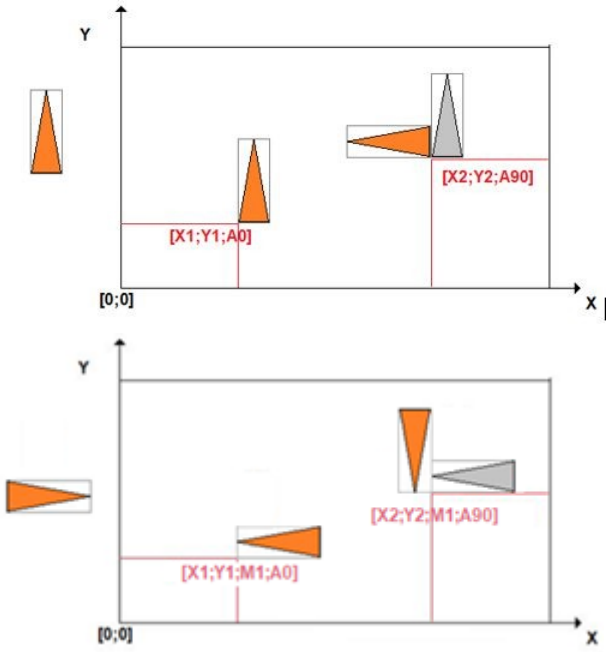
- *int ID*: identificativo numerico della *parte* corrispondente al piazzamento
- *int Item*: progressivo del piazzamento (>0)
- *double (X, Y)*: quote di piazzamento del vertice LB (inferiore sinistro) del rettangolo di ingombro originale della parte
- *short Mirror*: speculare richiesto per il piazzamento
  - 0 = none; 1 = x; 2 = y; 3 = x+y
  - l'asse per applicare la trasformata è passante per il centro del rettangolo di ingombro della parte
  - la trasformata è applicata prima della rotazione
- *double A*: angolo di rotazione, corrisponde al piazzamento (unità: gradi e decimali di grado)
  - il segno assegna la rotazione: positivo = antioraria; negativo = oraria
  - (X, Y) è il centro della rotazione
  - A assegna un angolo relativo: la parte ruota di A attorno a (X, Y)
- *bool InIsle*: true = piazzamento è in un'isola (solo in caso di nesting *True Shape*)
  - i due campi di tipo *int (InIsleMain, InIsleOut)* permettono di ricostruire per intero la progressione dei piazzamenti innestati in isole. Un client può ad esempio utilizzare un diverso ordine dei piazzamenti in modo da tagliare prima le parti piazzate entro le isole
- *string InCluster*: specifica se il piazzamento deriva dall'applicazione di un cluster
  - "": il piazzamento è singolo
  - altrimenti: deriva dall'esplosione di un cluster. Formato: "#;ID"
    - #: progressivo di applicazione di cluster (> 0). Piazzamenti con stesso valore derivano dallo stesso piazzamento di cluster
    - ID: identificativo numerico del cluster (campo *ID* di struttura *NestCluster*).
- *bool IsOver*: true = il piazzamento corrisponde ad un extra
- *bool IsInput*: true = corrisponde ad un piazzamento preliminare
  - il piazzamento preliminare di un cluster è comunque scorporato nel piazzamento delle singole parti.

I fattori che concorrono a determinare le trasformate di speculare e rotazione di un singolo piazzamento sono molteplici:
- assegnazione diretta della parte o del cluster di provenienza: campi (*Mirror*, *Rotate*) in strutture degli elementi
- settaggio *ModeMirrorPart*
- composizione geometrica in cluster

Le figure illustrano semplici casi di piazzamento di una parte con geometria regolare:

- la prima figura applica trasformate di traslazione e rotazione
- la seconda figura applica anche trasformata di speculare

Alla sinistra del sistema di assi coordinati è rappresentata la geometria originale della parte



ReadGeoInResult

La chiamata ricorsiva della funzione legge le geometrie che risultano assegnate per il piazzamento corrente.

bool ReadGeoInResult (int IndexItem, int IndexElement, ref NestGeometry Item)

Argomenti
- *IndexItem*: indice (base zero) sulla lista dei profili geometrici assegnati
- *IndexElement*: indice (base zero) dell'elemento geometrico in profilo geometrico
- *Item*: struttura di assegnazione dell'elemento geometrico

Valore di ritorno
True se il piazzamento sul foglio è individuato valido.

Note
La funzione deve essere chiamata subito dopo una chiamata di funzione ***ReadPartInResult*** e la prima chiamata deve utilizzare (*IndexItem=0, IndexElement=0*).

Un ritorno a questa prima chiamata *False* corrisponde ad una delle situazioni di errore:
- è in corso oppure non è valida una ottimizzazione di nesting
- non risulta piazzamento corrente valido.

La funzione permette di ricostruire la *forma* associata al piazzamento corrente, con applicate tutte le trasformate necessarie: traslazione, speculare, rotazione.

In caso di forma della parte assegnata come rettangolo puro (dimensioni in campi *L, H* della parte), la chiamata ricorsiva della funzione ritorna i quattro elementi lineari corrispondenti allo sviluppo del rettangolo.

SaveSolution

La funzione salva la *soluzione corrente* in file (formato XML).

NestErrors SaveSolution (string pathName)

Argomenti
- *pathName*: percorso del file

Valore di ritorno

NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorNoneSolution*) non risulta in presa una soluzione
- (*NestError.ErrorIOfile*) si è verificato un errore nell'accesso al file da scrivere.

Il file salvato può essere interpretato da un applicativo esterno in alternativa alla interrogazione delle funzioni.
Per i piazzamenti sui fogli, il file salva le informazioni basilari di ogni piazzamento come risultanti da struttura ***NestBox***.

SaveSolutionDXF

La funzione salva la *soluzione corrente* in uno o più file (formato DXF).

NestErrors SaveSolutionDXF (string pathName, bool autoClose, string layerSheet)

Argomenti
- *pathName*: percorso del file
- *autoClose*: true = genera profili corrispondenti alle parti sempre chiusi
- *layerSheet*: nome del livello di assegnazione del/i profilo/i di assegnazione del foglio.

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore già indicate per la funzione SaveSolution.

Se *pathName* assegna un'estensione differente da ".DXF", la stessa è aggiunta al nome completo del file.

La funzione salva un file per ogni foglio della *soluzione corrente* (fogli con piazzamenti).
Un foglio assegnato con ripetizioni è salvato una sola volta.
Il primo foglio è salvato con nome *pathName*. I restanti fogli modificano il nome originale con aggiunta di "_n", dove è n = progressivo (1, 2, ...).

Se *layerSheet* è assegnato significativo (es.: "sheet"):
- il file assegna anche la polilinea corrispondente alla definizione del foglio, con campo livello corrispondente alla stringa indicata
- ulteriori polilinee con stesso livello possono assegnare le isole interne al foglio
- le polilinee corrispondenti al foglio sono sempre riportate chiuse.

Se è *autoClose*=true: i profili corrispondenti alle parti sono riportati chiusi; altrimenti: come programmati.

3.14 Funzioni di serializzazione del progetto

Le funzioni gestiscono la serializzazione del progetto di nesting in/da file.

SaveProject

La funzione salva le assegnazioni delle liste di *parti, cluster* e *fogli* in file (formato XML).

NestErrors SaveProject (string pathName, bool bMode)

Argomenti
- *pathName*: percorso del file
- *bMode*: *true* richiede il salvataggio dei settaggi di funzionamento generale

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting

- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione di assegnazione (***IniSettings -> EndSettings***, ***IniSetPart -> EndSetPart***, ***IniSetSheet -> EndSetSheet***, ***IniSetCluster -> EndSetCluster***)
- (*NestError.ErrorIOProject*) si è verificato un errore nell'accesso al file da scrivere.

Utilizzare *bMode=true* per salvare anche i settaggi di funzionamento generale (corrispondenti a sezione ***IniSettings -> EndSettings***). L'utilità di un salvataggio che include anche i settaggi è evidente per situazioni di testing.
Dal salvataggio sono comunque esclusi alcuni settaggi (FixComputeError, TimerProgres, DirectoryTemp, RetrySquare) oltre ai piazzamenti preliminari.

LoadProject

La funzione legge le assegnazioni delle liste di *parti* e *fogli* da file (formato XML).

NestErrors LoadProject (string pathName, bool bMode)

Argomenti
- *pathName*: percorso del file
- *bMode*: *true* abilita la lettura dei settaggi di funzionamento generale

Valore di ritorno
NestError.ErrorNone se l'esito è positivo.

Note
Un ritorno differente da *NestError.ErrorNone* corrisponde ad una delle situazioni di errore:
- (*NestError.ErrorBusy*) è in corso una ottimizzazione di nesting
- (*NestError.ErrorContext*) la funzione è utilizzata entro una sezione di assegnazione (***IniSettings -> EndSettings***, ***IniSetPart -> EndSetPart***, ***IniSetSheet -> EndSetSheet***)
- (*NestError.ErrorIOProject*) si è verificato un errore nell'accesso al file da leggere.

Utilizzare *bMode=true* per leggere anche i settaggi di funzionamento generale (corrispondenti a sezione ***IniSettings -> EndSettings***): i settaggi che non sono letti dal file sono inizializzati ai valori di default. L'utilità di una lettura che includa anche i settaggi è evidente per situazioni di testing.
Dalla lettura sono comunque esclusi alcuni settaggi, che rimangono invariati: FixComputeError, TimerProgres, DirectoryTemp, RetrySquare.

Se *bMode=false*: i settaggi rimangono invariati.

I piazzamenti preliminari eventualmente assegnati sono azzerati.

4 Guida all'utilizzo della libreria

Scopo del capitolo è fornire le informazioni necessarie per l'integrazione della libreria TPA_N nella vostra applicazione.

La libreria TPA_N deve essere integrata come componente del vostro prodotto per un'architettura Windows a 32 e 64-bits.

4.1 Programma di installazione

Il programma di installazione può essere avviato in modalità normale o silenziosa.

Installazione normale

L'installazione in modalità normale crea un ambiente dimostrativo completo, a partire dalla cartella selezionata. La cartella di default è "C:\TpaNestingOEM".

L'installazione crea alcune sotto-cartelle:

- "\bin": per eseguibile dimostrativo (TpaNestingDEMO.exe) e librerie utilizzate

- "\help": per i tutorial della libreria TPA_N

- "\NestingOemCfg": cartella di configurazioni accessorie

- "\Examples": progetti di esempio per il nesting

- "\TestNesting": progetti dimostrativi di integrazione della libreria TPA_N (linguaggi di programmazione: C#, C, VbNET)

- "\HyLicenses": cartella per programma utile alla gestione della licenza software.

L'installazione normale è consigliata e necessaria per le valutazioni preliminari della libreria. L'applicativo dimostrativo fornisce un esempio di integrazione di TPA_N: l'ambiente, sviluppato in modo semplice e intuitivo, rende disponibile la quasi totalità delle opzioni previste, permettendo di valutarne l'effetto applicativo.

In cartella "\bin" si trova il file di configurazione dell'applicativo dimostrativo (TpaNestingDEMO.exe.config): contiene le informazioni necessarie per il funzionamento delle procedure di riconoscimento della licenza e può essere utilizzato come riferimento per la creazione del file equivalente per la vostra applicazione.

Nella cartella "\HyLicenses" c'è il file eseguibile HyLicenses.Exe: il programma gestisce l'installazione, l'aggiornamento ed il trasferimento della licenza. Per le modalità di utilizzo del programma si rimanda a documentazione dedicata.

Installazione silenziosa

L'installazione in modalità silenziosa (o nascosta) limita l'installazione ai soli file richiesti per l'integrazione della libreria nella vostra applicazione. Per *installazione silenziosa* si intende una installazione dove tutte le richieste che vengono normalmente fatte, come la cartella di destinazione o l'accettazione dei termini di licenza, non vengono mostrate e il tutto avviene in modo invisibile all'utilizzatore del computer.

L'utilizzo tipico dell'installazione silenziosa corrisponde al richiamo da procedura di installazione personalizzata.

L'avvio dell'installazione in modalità silenziosa riconosce l'assegnazione di comandi in riga di comandi

Comandi	*Significato*

/PATH /INSTALLDIR	Percorso completo di installazione del prodotto (es.: "/INSTALLDIR=c:\abc"). Direttamente nella cartella sono copiati tutti i file richiesti per l'integrazione della libreria
/HYLIC	Richiede l'installazione della cartella "\HyLicenses" (cartella per programma utile alla gestione della licenza software). La cartella è creata a partire da (/PATH)
/S	Richiede un'installazione silenziosa (il comando è richiesto in assenza di tutti gli altri).

4.2 Diagramma di flusso tipico

La sequenza riportata di seguito descrive un utilizzo di base della libreria:

1. creare una istanza della classe *TpaNestingOEM.Nesting()*
2. effettuare le verifiche preliminari di funzionamento generale (verifica della presenza della chiave e del livello di funzionamento abilitato: *Square* o *True Shape*)
3. registrazione delle funzioni di *CallBack*

4. apertura della sessione di assegnazione dei settaggi di funzionamento generale con chiamata della funzione *IniSettings()*
5. assegnazione dei settaggi di funzionamento generale (es.: *DirectoryTemp*, *TimeNesting*, …)
6. chiusura della sessione con chiamata della funzione *EndSettings()*

7. apertura della sessione di assegnazione delle *parti* con chiamata della funzione *IniSetPart ()*
8. assegnazione delle *parti* con chiamate della funzione *AddPart ()*: l'assegnazione di forme può avvenire in sessioni *IniGeometry ()*, …, *EndGeometry ()*
9. chiusura della sessione di assegnazione delle *parti* con chiamata della funzione *EndSetPart ()*

10. apertura della sessione di assegnazione dei cluster con chiamata della funzione *IniSetCluster ()*
11. assegnazione dei *cluster* con chiamate della funzione *AddCluster ()*: la composizione di un cluster avviene con chiamate a funzione *AddToCluster()*
12. chiusura della sessione di assegnazione dei *cluster* con chiamata della funzione *EndSetCluster()*

13. apertura della sessione di assegnazione dei *fogli* con chiamata della funzione *IniSetSheet ()*
14. assegnazione dei *fogli* con chiamate della funzione *AddSheet ()*
15. chiusura della sessione di assegnazione dei *fogli* con chiamata della funzione *EndSetSheet ()*

16. avvio della procedura di ottimizzazione con chiamata della funzione *Compute ()*
17. acquisizione dei risultati con chiamata alle funzioni: *ReadNumResult ()*, *ReadResult ()*, *ReadPartInResult ()*
18. acquisizioni/elaborazioni autonome delle informazioni di efficienza della procedura di nesting.

L'ordine di assegnazione di *settaggi*, *parti*, *cluster* e *fogli* è indifferente: quello proposto è solo indicativo. L'insieme di settaggi generali, parti, cluster e fogli costituiscono quello che è indicato come *progetto di nesting*, in quanto identifica l'insieme dei dati necessari all'esecuzione di una ottimizzazione.

Ordine delle funzioni di TPA_N

L'ordine di assegnazione di *settaggi*, *parti*, *cluster* e *fogli* è indifferente: quello proposto è solo indicativo, ma è anche certo che corrisponde al flusso logico delle informazioni.
Come già ampiamente visto, le chiamate di funzioni di ogni gruppo di assegnazione devono rispettare una determinata sequenza. Per garantire la sequenza corretta, quasi tutte le funzioni eseguono controlli di livello di ingresso per scoprire se la libreria è in uno stato corretto al momento della ricezione della chiamata. Se la libreria non viene trovata nello stato corretto, le funzioni restituiscono un errore indicativo della situazione.

Si consiglia pertanto agli utenti di controllare i valori di ritorno di ciascuna funzione, in modo da individuare situazioni non corrette direttamente in fase di collaudo della propria applicazione.

L'insieme di settaggi generali, parti, cluster e fogli costituisce quello che è indicato come *progetto di nesting*, in quanto identifica l'insieme dei dati necessari all'esecuzione di una ottimizzazione.

4.3 Verifiche preliminari

Una verifica preliminare della presenza della chiave, del livello di funzionamento abilitato e delle opzioni aggiuntive può essere utile e necessario per adattare le funzionalità del vostro applicativo:
- verificare la presenza e validità della chiave con interrogazione di IsSquareEnabled
- verificare il livello di funzionamento avanzato di Tpa_N con interrogazione di IsShapeEnabled
- con verificato il livello di funzionamento avanzato, verificare le opzioni disponibili in lettura di disegno da file esterno, con interrogazione di ShapeExtension.

4.4 Assegnazione dei settaggi

È generalmente necessario effettuare una assegnazione preliminare dei settaggi di funzionamento generale:
- la prima funzione che deve essere chiamata è *IniSettings(...)*
- l'ultima funzione che deve essere chiamata è *EndSettings(...)*.

Tutte le proprietà relative all'assegnazione dei settaggi devono essere utilizzate tra le due funzioni citate sopra.
Si rammenta che la lettura degli stessi settaggi può essere effettuata anche in contesto differente.
L'assegnazione preliminare dei settaggi può anche avvenire con lettura da un file salvato in precedenza: vedi funzioni *SaveSettings()*, *LoadSettings()*.

Una parte dei settaggi di funzionamento generale è di certo più frequentemente modificata, in quanto direttamente correlata ad un *progetto di nesting*: è l'applicazione esterna che decide quale organizzazione dare ai settaggi e con quale criterio e frequenza aggiornarli.

Una modifica dei settaggi inibisce la possibilità di effettuare nuovi tentativi di ottimizzazione: una successiva chiamata a funzione RetryCompute fallisce.

4.5 Assegnazione delle parti

Si passa all'assegnazione della lista di parti:
- la prima funzione che deve essere chiamata è *IniSetPart()*
- l'ultima funzione che deve essere chiamata è *EndSetPart()*.

Se l'assegnazione della lista di parti è totale, chiamare *IniSetPart* con argomento *Clear=true*.
Una modifica delle parti inibisce la possibilità di effettuare nuovi tentativi di ottimizzazione (una successiva chiamata a funzione RetryCompute fallisce).

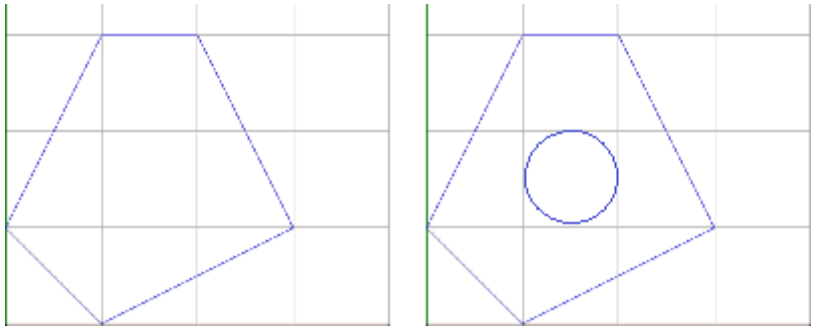
È opportuno sottolineare alcuni punti fondamentali nell'assegnazione delle parti:
- ogni parte ha un identificativo numerico univoco, strettamente positivo (> 0): campo *ID* in struttura *NestPart*
- non è quindi possibile assegnare lo stesso *ID* per più parti
- gli identificativi possono essere assegnati in ordine qualunque e non necessariamente consecutivi
- occorre tenere presente che gli ID hanno un ruolo importante per l'ordinamento delle parti: a parità di altre impostazioni notevoli (Es.: priorità, geometria della parte) è l'ID che decide quale parte ha una maggiore priorità di piazzamento nel processo di ottimizzazione (con corrispondenza inversa: ad ID inferiore corrisponde maggiore priorità).

Per aggiungere una parte chiamare la funzione *AddPart()*.
Per eliminare una parte chiamare la funzione *RemovePart()* con argomento *OnlyGeometry=false*.
Per modificare una parte chiamare la funzione *WritePart()*.
Per modificare la *forma* di una parte già inserita chiamare la funzione *RemovePart()* con argomento *OnlyGeometry=true*.

Come assegnare la forma di una parte

Per una *parte* già assegnata è possibile definire una forma.
L'assegnazione delle forme di parti deve avvenire entro una sezione (***IniSetPart -> EndSetPart***).

In figura due esempi di forme:
- a sinistra una forma semplice con assegnata la sola geometria esterna
- a destra una forma più complessa con assegnata anche una geometria interna (isola).



Ogni geometria viene assegnata in modo indipendente:
- la prima funzione che deve essere chiamata è *IniGeometry()*
- l'ultima funzione che deve essere chiamata è *EndGeometry()*.

La funzione *IniGeometry* indica:
- su quale parte effettuare l'assegnazione
- se si tratta di geometria esterna o interna.

Per ogni parte può essere assegnata una sola geometria esterna, prima di quelle interne.
Per una parte è possibile assegnare solo geometrie interne: quella esterna è assegnata in automatico di forma rettangolare, applicando le dimensioni impostate per la parte e posizionando lo spigolo inferiore sinistro sull'origine degli assi coordinati.

Codice di esempio

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

...
NestPart Item=new NestPart();

//apre la sezione di assegnazione delle parti
If (nestObj.IniSetPart(true))
{
    //assegnazione minima della struttura della parte (ID=1, N=20)
    Item.Initialize (1);
    Item.N=20;
    //aggiunge la parte
    nestObj.AddPart(Item);
    //apre la sezione di assegnazione della geometria esterna della parte di ID=1
    nestObj.IniGeometry(1, false, 0.0, 100);
    nestObj.AddToGeometry_Line(100, 300);
    nestObj.AddToGeometry_Line(200, 300);
    nestObj.AddToGeometry_Line(300, 100);
    nestObj.AddToGeometry_Line(100, 0);
    //chiude la sezione di assegnazione della geometria (esterna)
    nestObj.EndGeometry();

    //apre la sezione di assegnazione della geometria interna della parte di ID=1
    nestObj.IniGeometry(1, true, 100, 150);
    nestObj.AddToGeometry_Circle(150, 150);
    //chiude la sezione di assegnazione della geometria (interna)
    nestObj.EndGeometry();

    //aggiunge le restanti parti
    ...
    //chiude la sezione di assegnazione delle parti
    nestObj.EndSetPart();
```

```
}
```

Codice di esempio: ciclo di acquisizione di geometria esterna

```
bool GeometryFromFile (string pathName)
{
    //errore da gestire al ritorno della funzione
    if (nestObj.ReadShape (pathName) != NestErrors.ErrorNone) return false;

    //inizializza le variabili utilizzate sul ciclo di lettura
    int IndexItem = 0;              //indice delle geometrie
    int IndexElement = 0;           //indice dell'elemento di geometria (IndexItem)
    NestGeometry Item=new NestGeometry();

    //ciclo sulle geometrie acquisite
    while (nestObj.ReadGeoInShape (IndexItem, IndexElement, ref Item))
    {
        // ……tratta (Item)=1' elemento della geometria di indice (IndexItem)

        IndexElement++;             //incremento indice dell'elemento di geometria (IndexItem)
        while (nestObj. ReadGeoInShape (IndexItem, IndexElement, ref Item))
        {
            // ……tratta (Item)= nuovo elemento della geometria di indice (IndexItem)

            IndexElement++;         //incremento indice dell'elemento di geometria (IndexItem)
        }
        IndexItem ++;               // incrementa l'indice delle geometrie
        IndexElement=0;             //inizializza indice dell'elemento di geometria (IndexItem)
    }
    return true;
}
```

4.6 Assegnazione dei fogli

Si passa all'assegnazione della lista dei fogli:
- la prima funzione che deve essere chiamata è *IniSetSheet()*
- l'ultima funzione che deve essere chiamata è *EndSetSheet()*.

Se l'assegnazione della lista di fogli è totale, chiamare *IniSetSheet* con argomento *Clear=true*.
Una modifica dei fogli inibisce la possibilità di effettuare nuovi tentativi di ottimizzazione (una successiva chiamata a funzione RetryCompute fallisce).

Come già per le parti, è opportuno sottolineare alcuni punti fondamentali nell'assegnazione dei fogli:
- ogni foglio ha un identificativo numerico univoco, strettamente positivo (> 0): campo *ID* in struttura *NestSheet*
- non è quindi possibile assegnare lo stesso *ID* per più fogli
- gli identificativi possono essere assegnati in ordine qualunque e non necessariamente consecutivi
- gli ID hanno un ruolo importante per l'ordinamento dei fogli: a parità di altre impostazioni notevoli (Es.: priorità, informazione di sfrido) è l'ID che decide quale foglio ha una maggiore priorità di utilizzo nel processo di ottimizzazione.

Per aggiungere un foglio chiamare la funzione *AddSheet()*.
Per eliminare un foglio chiamare la funzione *RemoveSheet()* con argomento *OnlyGeometry=false*.
Per modificare un foglio chiamare la funzione *WriteSheet()*.
Per modificare la *forma* di un foglio già inserita chiamare la funzione *RemoveSheet()* con argomento *OnlyGeometry=true*.

Come assegnare la forma di un foglio

Per un *foglio* già assegnato è possibile definire una forma.
L'assegnazione delle forme di *fogli* deve avvenire entro una sezione (***IniSetSheet -> EndSetSheet***).

Le modalità di assegnazione ricalcano quanto già esposto per le *parti*.

4.7 Assegnazione degli abbinamenti manuali

Si passa all'assegnazione (opzionale) della lista dei cluster:
- la prima funzione che deve essere chiamata è *IniSetCluster()*
- l'ultima funzione che deve essere chiamata è *EndSetCluster()*.

Se l'assegnazione della lista di cluster è totale, chiamare *IniSetCluster* con argomento *Clear=true*.
Una modifica dei cluster inibisce la possibilità di effettuare nuovi tentativi di ottimizzazione (una successiva chiamata a funzione RetryCompute fallisce).

È opportuno sottolineare alcuni punti fondamentali nell'assegnazione dei cluster:
- ogni cluster ha un identificativo numerico univoco, strettamente positivo (> 0): campo *ID* in struttura *NestCluster*
- non è quindi possibile assegnare lo stesso *ID* per più cluster
- gli identificativi possono essere assegnati in ordine qualunque e non necessariamente consecutivi
- gli ID hanno un ruolo importante per l'ordinamento dei cluster: a parità di altre impostazioni notevoli (Es.: priorità, composizione del cluster) è l'ID che decide quale cluster ha una maggiore priorità di piazzamento nel processo di ottimizzazione.

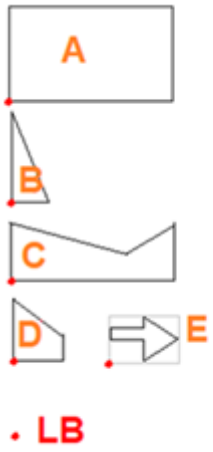
Per aggiungere un cluster chiamare la funzione *AddCluster()*.
Per definire la composizione di un cluster chiamare la funzione *AddToCluster()*.

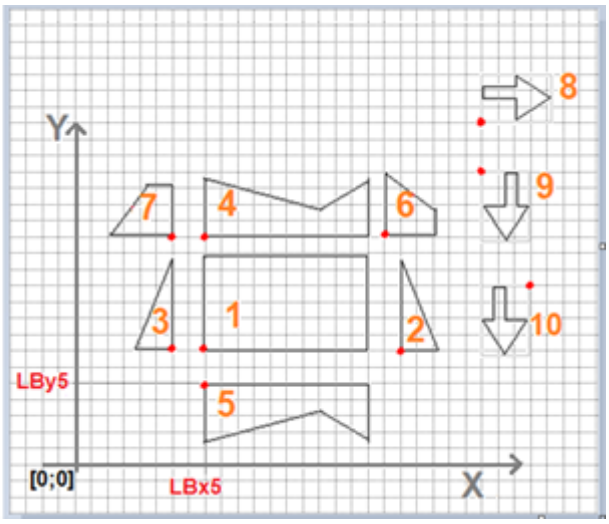
Per eliminare un cluster chiamare la funzione *RemoveCluster()* con argomento *OnlyGeometry=false*.
Per modificare un cluster chiamare la funzione *WriteCluster()*.
Per modificare la composizione di un cluster già inserito chiamare la funzione *RemoveCluster()* con argomento *OnlyGeometry=true*.

Come assegnare la composizione di un cluster

Per un *cluster* già assegnato è possibile definire una composizione.
L'assegnazione della composizione di un cluster deve avvenire entro una sezione (***IniSetCluster -> EndSetCluster***).
La composizione di un cluster è definita con chiamate a metodo ***AddToCluster***.

In figura un esempio delle regole di costruzione dei cluster:
- a sinistra vediamo le parti piazzabili
  - le lettere (A, B, C, D, E) possono corrispondere al nome descrittivo delle parti (campo *Label* in struttura *NestPart*)
  - il cerchietto in colore rosso che è riportato su ogni parte è il punto LB (left-bottom) del rettangolo di ingombro delle parti
  - solo per la parte (E) il punto LB non è sul profilo: attorno ad (E) è riportato il rettangolo di ingombro

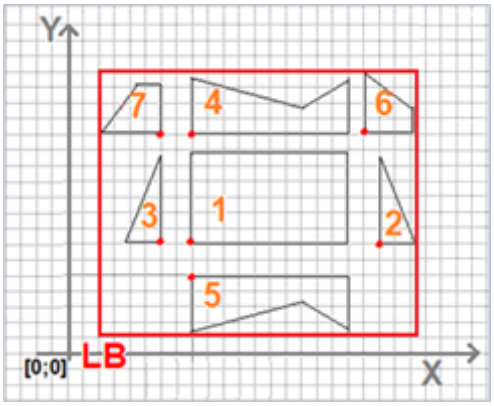




- a destra sono riportati possibili utilizzi delle parti in un cluster
  - sono riportate 10 parti (numerate da 1 a 10)
  - la tabella riporta le informazioni di costruzione del cluster (vedi di seguito: Codice di esempio):

#	*Parte*	*X*	*Y*	*Mirror*	*A°*
1	A	LBx1	LBy1	0	0
2	B	LBx2	LBy2	0	0
3	B	LBx3	LBy3	x	0
4	C	LBx4	LBy4	0	0
5	C	LBx5	LBy5	y	0
6	D	LBx6	LBy6	0	0
7	D	LBx7	LBy7	0	90
8	E	LBx8	LBy8	0	0
9	E	LBx9	LBy9	0	-90
10	E	LBx10	LBy10	x	90

- con costruzione del cluster sui primi 7 elementi, la figura riporta
  - il rettangolo di ingombro del cluster (riquadro di colore rosso)
  - il punto LB del cluster



Codice di esempio

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

...
NestCluster Item=new NestCluster();
ItemCluster OneItem=new ItemCluster();


//apre la sezione di assegnazione dei cluster
If (nestObj.IniSetCluster(true))
{
  //assegnazione minima della struttura del cluster (ID=1, N=20)
  Item.Initialize (1);
  Item.N=20;
  //aggiunge un cluster
  nestObj.AddCluster(Item);
  //assegna la composizione del cluster
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "A", X = 160.0, Y = 140.0, Mirror = 0, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "B", X = 400.0, Y = 140.0, Mirror = 0, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "B", X = 120.0, Y = 140.0, Mirror = 1, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "C", X = 160.0, Y = 280.0, Mirror = 0, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "C", X = 160.0, Y = 100.0, Mirror = 2, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "D", X = 380.0, Y = 280.0, Mirror = 0, A =
  0.0});
  nestObj.AddTocluster(Item.ID, new ItemCluster() { NameID = "D", X = 120.0, Y = 280.0, Mirror = 0, A =
  90.0});
  ...

  //aggiunge i restanti cluster
  ...
  //chiude la sezione di assegnazione dei cluster
  nestObj.EndSetCluster();
}
```

4.8 Assegnazione dei piazzamenti preliminari

Si passa all'assegnazione (opzionale) della lista dei piazzamenti preliminari:
- la prima funzione che deve essere chiamata è *IniSetPlaced()*
- l'ultima funzione che deve essere chiamata è *EndSetPlaced()*.

Se l'assegnazione della lista è totale, chiamare *IniSetPlaced* con argomento *Clear=true*.
Una modifica dei piazzamenti preliminari inibisce la possibilità di effettuare nuovi tentativi di ottimizzazione (una successiva chiamata a funzione RetryCompute fallisce).

Per aggiungere un piazzamento chiamare la funzione *AddPlaced()*.
Per eliminare un piazzamento chiamare la funzione *RemovePlaced()*.

Codice di esempio

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

...
NestPlaced Item=new NestPlaced ();
```

```
//apre la sezione di assegnazione dei piazzamenti preliminari
If (nestObj.IniSetPlaced(true))
{
   //primo piazzamento su primo foglio (cluster di ID=3)
   Item.Initialize (); Item.Type = true; Item.ID = 3;
   Item.X = 30.0; Item.Y = 50.0; Item.A = 0.0; Item.Mirror = 0;
   nestObj.AddPlaced(1, item);
  //secondo piazzamento su primo foglio (parte di ID=1)
   Item.Initialize (); Item.ID = 1;
   Item.X = 420.0; Item.Y = 50.0; Item.A = 30.0; Item.Mirror = 0;
   nestObj.AddPlaced(1, item);

   //aggiunge i restanti piazzamenti
   …
   //chiude la sezione di assegnazione dei piazzamenti preliminari
   nestObj.EndSetPlaced();
}
```

4.9 Esecuzione dell'ottimizzazione di nesting

Per avviare una ottimizzazione del progetto di nesting è necessario chiamare la funzione Compute.
Prima dell'ottimizzazione sono eseguite verifiche sulle parti e sui fogli che possono anche determinare un ritorno della funzione con errore e conseguente annullamento dell'ottimizzazione:

- devono risultare parti o cluster richiesti per i piazzamenti e fogli disponibili
- la verifica dei filtri di corrispondenza (spessore, materiale, colore) deve almeno potere abbinare una parte o un cluster ad un foglio.

Durante una ottimizzazione l'applicazione esterna può gestire le funzioni di CallBack ed eventualmente richiedere una interruzione dell'ottimizzazione stessa.

Acquisire il risultato della soluzione

Ad ottimizzazione eseguita è possibile acquisire i risultati della soluzione di nesting.

Alcune proprietà/metodi restituiscono informazioni generali relative alla procedura di ottimizzazione:
- *IsComputed*: stato di ottimizzazione valida
- *ModeCompute*: informazione dell'ottimizzazione eseguita (*Square* o *True Shape*)
- *TimeOut*: informazione se la procedura di calcolo è terminata per raggiunto time-out
- *CanRetry*: possibilità di richiedere una nuova ottimizzazione.

Un gruppo di funzioni permette invece di acquisire tutte le informazioni relative alla soluzione dell'ottimizzazione:
- *Fitness*: efficienza della soluzione
- *ReadNumResult*: informazione del numero di fogli complessivi della soluzione o di un tipo di foglio
- *ReadNumPartInResult*: informazione del numero di piazzamenti della soluzione o di un foglio e/o di un tipo di parte
- *ReadNumClusterInResult*: informazione del numero di piazzamenti della soluzione o di un foglio e/o di un tipo di cluster
- *ReadResult*: informazioni generali relative a un singolo foglio della soluzione
- *ReadPartInResult*, *ReadGeoInResult*: informazioni di un singolo piazzamento su un foglio della soluzione
- *SaveSolution*: salva la soluzione su file (formato XML)
- *SaveSolutionDXF*: salva la soluzione su uno o più file (formato DXF).

Codice di esempio: come acquisire le informazioni generali dei fogli della soluzione

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

…
//Campi di assegnazione su interrogazione del componente
int ID_Sheet = 0, Item_Sheet = 0, NumPlaces = 0, Repetition= 0;
double AreaPercent = 0.0;
```

```
//ciclo di interrogazioni sui fogli della soluzione
int IndexSheet = 0; //indice di foglio
while (nestObj.ReadResult (IndexSheet, ref ID_Sheet, ref Item_Sheet, ref AreaPercent, ref NumPlaces, ref
Repetition))
{
    // ...
    //ciclo di acquisizione dei piazzamenti di un foglio
    //.....
    IndexSheet ++; // incrementa l'indice del foglio
}
```

Codice di esempio: ciclo di acquisizione dei piazzamenti di un foglio

```
// IndexSheet = indice di foglio (vedi: ciclo precedente)
int IndexBox = 0; //indice di piazzamento sul foglio
NestBox nestBox=new NestBox();

while (nestObj.ReadPartInResult (IndexSheet, IndexBox, ref nestBox))
{
    // .....
    // vedi: "Ciclo di acquisizione della geometria di un piazzamento"
    //.....

    IndexBox++; // incrementa l'indice del piazzamento
    // .....
}
```

Codice di esempio: ciclo di acquisizione della geometria di un piazzamento

Il ciclo di lettura deve essere eseguito dopo una chiamata a funzione ***ReadPartInResult***.

```
int IndexItem = 0;      //indice delle geometrie della parte
int IndexElement = 0;   //indice dell'elemento di geometria (IndexItem)
NestGeometry Item=new NestGeometry();

//ciclo sulle geometrie della parte
while (nestObj.ReadGeoInResult (IndexItem, IndexElement, ref Item))
{
    // .....tratta (Item) = 1' elemento della geometria di indice (IndexItem)

    IndexElement++;             //incremento indice dell'elemento di geometria (IndexItem)
    while (nestObj.ReadGeoInResult (IndexItem, IndexElement, ref Item))
    {
        // .....tratta (Item)= nuovo elemento della geometria di indice (IndexItem)

        IndexElement++;         //incremento indice dell'elemento di geometria (IndexItem)
    }
    IndexItem ++;       // incrementa l'indice delle geometrie
    IndexElement=0;     //inizializza indice dell'elemento di geometria (IndexItem)
}
```

Calcolare e gestire soluzioni progressive

Con ottimizzazione *True Shape* è possibile calcolare più soluzioni, fino ad un massimo di 20:

- la proprietà CanRetry ottiene l'informazione se è possibile eseguire una nuova ottimizzazione
- chiamare la funzione RetryCompute per eseguire la nuova ottimizzazione.

Ad ottimizzazione eseguita, la soluzione calcolata è impostata in automatico come attuale.

Tutte le soluzioni calcolate rimangono disponibili, con possibilità di navigare tra le medesime cambiando la soluzione corrente su una specifica.

La soluzione corrente può essere salvata su file chiamando la funzione SaveSolution (o SaveSolutionDXF).

Codice di esempio: navigazione tra soluzioni progressive

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

// GoToNextSolution: passa alla soluzione successiva calcolata
bool GoToNextSolution()
{
    If (nestObj.Solution < nestObj.OfSolution)
    {
        nestObj.Solution = nestObj.Solution+1;
        return true;
    }
    return false;
}

// GoToPrevSolution: passa alla soluzione precedente calcolata
bool GoToPrevSolution()
{
    If (nestObj.Solution >1)
    {
        nestObj.Solution = nestObj.Solution-1;
        return true;
    }
    return false;
}
```

4.10 Processare le funzioni di CallBack

Le funzioni sono utilizzate per monitorare l'avanzamento della fase di ottimizzazione. Al momento è disponibile una sola funzione.

Funzioni Progres

La gestione dell'evento permette di annullare o interrompere la procedura di ottimizzazione:
- l'annullamento determina la chiusura della fase di ottimizzazione con cancellazione totale della procedura
- l'interruzione determina la chiusura della fase di ottimizzazione al completamento della prima soluzione valida.

```
// Funzione Progres
void ProgresFromNesting (int nValue, ref bool bCancel, ref bool bPause)
{
    //Aggiorna una finestra di avanzamento
    ...
    // Gestione interattiva su selezioni utente:
    // CancelCondition=true -> condizione di interruzione dell'ottimizzazione
    // StopCondition=true -> condizione di uscita veloce dell'ottimizzazione

    If (CancelCondition) bCancel=true;
    else if (StopCondition) bPause=true;

}
```

4.11 Salvare e leggere un progetto di TPA_N

È possibile salvare e recuperare un progetto mediante le funzioni SaveProject e LoadProject.

Le due funzioni permettono di salvare e recuperare anche i settaggi di funzionamento generale.
La possibilità di richiedere il salvataggio di un progetto che includa anche i settaggi generali può essere utile per scopi di test, nel caso in cui sia appunto richiesto di testare ripristinando una situazione critica.

Dopo l'esecuzione di *LoadProject*, un applicativo esterno deve aggiornare parti cluster e fogli, con lettura delle informazioni da Tpa_N.

Codice di esempio

```
TpaNestingOEM.Nesting nestObj=new TpaNestingOEM.Nesting();

//Strutture
NestPart ItemPart=new NestPart();
NestSheet ItemSheet=new NestSheet();
NestSize ItemSize=new NestSize();
NestGeometry ItemGeo=new NestGeometry();
NestCluster ItemCluster=new NestCluster();
ItemCluster OneItemCluster=new ItemCluster();

// Legge il progetto
if (nestObj.LoadProject ("C:\projects\one.xml", false) == NestErrors.ErrorNone)
{
    //---------- Legge le parti
    for (int idxElement=0; idxElement< nestObj.CountPart; idxElement++)
    {
        nestObj.ReadPartIndex (idxElement, ref ItemPart, ref ItemSize);

        int IndexItem = 0;          //indice delle geometrie della parte
        int IndexElement = 0;       //indice dell'elemento di geometria (IndexItem)

        //ciclo sulle geometrie della parte
        while (nestObj.ReadGeometry (0, ItemPart.ID, IndexItem, IndexElement, ref ItemGeo)) ==
                                                                    NestErrors.ErrorNone)
        {
            // ……tratta (ItemGeo) = 1' elemento della geometria di indice (IndexItem)

            IndexElement++;       //incremento indice dell'elemento di geometria (IndexItem)
            while (nestObj ReadGeometry (0, ItemPart.ID, IndexItem, IndexElement, ref ItemGeo)) ==
                                                                NestErrors.ErrorNone)
            {
                // ……tratta (ItemGeo)= nuovo elemento della geometria di indice (IndexItem)

                IndexElement++;  //incremento indice dell'elemento di geometria (IndexItem)
            }
            IndexItem ++;                   // incrementa l'indice delle geometrie
            IndexElement=0;                 //inizializza indice dell'elemento di geometria (IndexItem)
        }
    }


    //---------- Legge i fogli

    for (int idxElement=0; idxElement< nestObj.CountSheet; idxElement++)
    {
        nestObj.ReadSheetIndex (idxElement, ref ItemSheet, ref ItemSize);
        int IndexItem = 0;          //indice delle geometrie del foglio
        int IndexElement = 0;       //indice dell'elemento di geometria (IndexItem)
```

```
        //ciclo sulle geometrie del foglio
        while (nestObj.ReadGeometry (1, ItemShet.ID, IndexItem, IndexElement, ref ItemGeo)) ==
                                                                    NestErrors.ErrorNone)
        {
            // ...
        }
    }

    // Legge i cluster
    for (int idxElement=0; idxElement< nestObj.CountCluster; idxElement++)
    {
        nesObj.ReadClusterIndex (idxElement, ref ItemCluster);

        int IndexItem = 0;          // indice delle assegnazioni del cluster

        //ciclo sulle assegnazioni del cluster
        while (nestObj.ReadInCluster ( ItemCluster.ID, IndexItem, ref OneItemCluster) ==
        NestErrors.ErrorNone)
        {
            // ...
            IndexItem ++;           // incrementa l'indice delle assegnazioni
        }
    }
}
```

Tecnologie e Prodotti per l'Automazione S.r.l.

Via Carducci 221
20099 Sesto S.Giovanni (Milano)
ITALY
Tel. +39 0236527550

www.tpaspa.com

info@tpaspa.it

3/6/2026