# N-QUEENS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***value ordering heuristics of Gecode*** | ***n*** | | | |
| *30* | *35* | *45* | *50* |
| ***number of failures*** | | | |
| input order – min value | 1,588,827 | 2,828,740 | - | - |
| input order – random value | 9 | 10 | 6 | 42 |
| min domain size – min value | 15 | 21 | 6 | 123 |
| min domain size – random value | **1** | **0** | **1** | **10** |
| domWdeg – min value | 15 | 21 | 6 | 123 |
| domWdeg – random value | **1** | **0** | **1** | **10** |

Tests were conducted with different values of N using various variable and value ordering heuristics in the context of the n-regine problem solved with the Gecode solver. The table shows the numbers of failures, with the best results highlighted in bold.

## Analysis and comparison of heuristics

Nella tabella si può notare che Input Order – Min Value presenta un numero elevato di fallimenti, il che indica una ricerca meno efficiente.

La scelta di selezionare le variabili nell'ordine dato e di testare prima il valore più basso ha il risultato peggiore in termini di fallimenti. Il motivo è che la selezione delle variabili nell'ordine dell'input costringe a non avere scelta su quali variabili sia meglio selezionare per prime.

Pertanto, se la prima decisione è sbagliata e il solutore finisce in un sottoalbero di grandi dimensioni, può richiedere molto tempo per uscirne e quindi presenta tempi di esecuzione prolungati.

In effetti, per alcune istanze, il risolutore impiega molto più tempo a trovare una soluzione, probabilmente a causa di un particolare ordinamento delle variabili. Inoltre, notiamo che per le istanze in cui n è 45 e 50, il risolutore non è in grado di trovare una soluzione entro il tempo richiesto (5 minuti).

Notando complessivamente la tabella e mettendo a confronto le euristiche con min value e quelle con random value notiamo che utilizzando la randomizzazione i risultati migliorano. Esaminando diverse parti dell'albero, aumenta la probabilità di trovare una soluzione, riducendo così il rischio di rimanere bloccati in un sottoalbero.

Inoltre, notiamo che le due euristiche min domain size e dowWdeg sia con i min value che con i random value presentano gli stessi fallimenti.

Questo si manifesta poiché ogni variabile inizialmente comprende tutti i vincoli, portando così a pesi inizialmente equivalenti per tutte le variabili. Quando il risolutore si scontra con un fallimento, il peso associato al vincolo viene incrementato. Tuttavia, dal momento che tutte le variabili condividono vincoli identici, il peso di ciascuna variabile aumenta simultaneamente di 1. Di conseguenza, i pesi rimangono costanti durante l'intera esecuzione, e l'unico determinante nella selezione delle variabili diventa la dimensione del dominio.

In breve, possiamo dire che il criterio di scelta dei valori si basa sul rapporto tra la dimensione del dominio e il grado ponderato di ogni variabile nella scelta della variabile. Questo criterio favorisce la selezione di variabili con domini più piccoli e gradi ponderati più alti.

QUESTO MI SA CHE L’HO PRESO DELLA TEORIA

Filtering variables by taking only the smallest domain allows one to dynamically choose the variable with the smallest domain size and dynamically adjust the search according to the domain size after propagation.

Dynamic heuristics not only compares with propagation, but also with search.

* At each step we choose the variable according to the size of the domain after propagation.
* More specifically, the variable with the smallest domain is selected at each step. Theoretically, dynamic heuristics are implemented during the search, thus making it more onerous.

QUESTO FACEVA PARTE DELLE CONCLUSIONI FINALI NON SO DOVE METTERLO E SE METTERLO, CREDO DI AVERLO PRESO DALLA TEORIA

Quando si esegue la Backtracking Tree Search - BTS, la propagazione viene intercalata per ridurre la dimensione esponenziale dell'albero di ricerca. Questa tecnica consente alla ricerca di concentrarsi sulle porzioni più promettenti dello spazio delle soluzioni, accelerando il processo di risoluzione del problema.

# POSTER PLACEMENT

## ORIGINAL ORDER

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***value ordering heuristics of Gecode*** | Data files | | | | | |
| 19x19 | | | 20x20 | | |
| ***number of failures*** | ***time*** | ***number of failures*** | | ***time*** |
| input order – min value | 1,362,457 | 2s 962msec | - | | - |
| input order – random value | - | - | - | | - |
| min domain size – min value | 239,954 | 4s 718msec | **1,873** | | **446msec** |
| min domain size – random value | 2,929,153 | 55s 830msec | 5,797,312 | | 1m 42s |
| domWdeg – min value | **236,024** | **6s 25msec** | **1,873** | | **483msec** |
| domWdeg – random value | 2,929,030 | 54s 109msec | 5,797,456 | | 1m 38s |

## ORDER BY DECREASING PERIMETER

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***value ordering heuristics of Gecode*** | Data files | | | |
| 19x19 | | 20x20 | |
| ***number of failures*** | ***time*** | ***number of failures*** | ***time*** |
| input order – min value | **30** | **176msec** | **323** | **183msec** |
| input order – random value | - | - | - | - |

We calculated the results we entered in the table with the original order of the values and then reordered the rectangles in the data file in descending order according to their perimeter.

Analysis and comparison of heuristics

L'euristica di ordinamento degli input, denominata "input order", mostra tempi di esecuzione lunghi e fallimenti frequenti. Questa tecnica si dimostra poco flessibile e incontra difficoltà nell'identificare le variabili vantaggiose nelle prime fasi della ricerca.

Come il problema precedente, la prima tabella evidenzia risultati praticamente equivalenti tra le euristiche "valore minimo" e "domWdeg", sebbene quest'ultima produca talvolta meno fallimenti. Questa disparità è attribuita alla presenza di vincoli diversi per i rettangoli. In particolare, collocare rettangoli più grandi risulta più complesso a causa di vincoli più stringenti rispetto a quelli associati a rettangoli più piccoli. Questa variazione nei vincoli e nelle dimensioni dei rettangoli contribuisce a una differenza significativa, poiché i rettangoli più difficili da posizionare generano guasti più frequenti a causa della maggiore rigidità dei loro vincoli e del conseguente peso maggiore. Pertanto, essi sono preferiti nelle scelte iniziali. In alcuni scenari, questa dinamica rende l'euristica "domWdeg" leggermente più efficace del criterio del "dominio più piccolo", anche se entrambi sono superati dall'euristica statica quando i dati sono riordinati.

Nel contesto di questo problema, l'applicazione di casualità non conduce mai a miglioramenti, anzi, aumenta il numero di fallimenti. La posizione dei poster è fortemente influenzata dai vincoli stringenti del problema. Di conseguenza, l'utilizzo di un approccio deterministico, che segue i vincoli, produce risultati migliori rispetto alla randomizzazione.

Nella seconda tabella, dopo il riordinamento decrescente dei poster, si osserva una significativa diminuzione dei fallimenti nella prima euristica di ordinamento delle variabili. Questo si verifica perché i primi poster presentano vincoli più stringenti, e posizionarli inizialmente è vantaggioso. In caso di fallimento, questo avviene immediatamente, evitando che i vincoli successivi causino ulteriori fallimenti dopo il posizionamento di altri poster. Allo stesso modo, i poster più piccoli con vincoli meno stringenti sono più facilmente posizionabili in un secondo momento dell'esecuzione.

Nonostante l'aggiunta di casualità continui a non produrre miglioramenti, arrivando addirittura al superamento del tempo limite nella ricerca di questa euristica.

Possiamo concludere affermando che il riordino decrescente dei rettangoli rappresenta la caratteristica principale e più rilevante di "Poster Placement", poiché consente di programmare la ricerca per ottenere prestazioni migliori.

QUESTO SPERA SIA GIUSTO HO COPIATO LE COSE Più PARTICOLARI DA RICCARDO perché IO NON LE AVEVO SCRITTE

# QUASIGROUP COMPLETION PROBLEM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data files |  | default search | domWdeg – random value | domWdeg – random value + restarting |
| *30-03* | ***number of failures*** | - | 1,061,184 | **642,427** |
| ***time*** | - | 2m 38s | **1m 42s** |
| *30-05* | ***number of failures*** | 657,955 | **5,885** | 303,205 |
| ***time*** | 1 m 33s | **1s 217 msec** | 45s 627msec |
| *30-08* | ***number of failures*** | **627** | 6,403 | 11,990 |
| ***time*** | **339msec** | 1s 180msec | 2s 454msec |
| *30-12* | ***number of failures*** | 259,082 | 53,200 | **21,986** |
| ***time*** | 32s 517msec | 8s 0msec | **3s 867msec** |
| *30-19* | ***number of failures*** | 381,330 | - | **48,244** |
| ***time*** | 54s 790msec | - | **8s 880msec** |

The table above shows the number of failures and the total time required by different search types for the quasigroup completion problem.

## Analysis and comparison of heuristics

L’unico caso in cui il metodo di ricerca predefinito è più efficiente è il caso qc30\_08, dove il solutore governa completamente la ricerca di default. Probabilmente alcune istanze della ricerca predefinita sono influenzate da alcune decisioni sbagliate prese all'inizio dell'albero. La ricerca predefinita dipende completamente dal risolutore, quindi potrebbe accadere che prenda una decisione sbagliata all'inizio della ricerca, si blocchi lì e richieda molto tempo di ricerca per annullarla. Invece, in questo caso le prime assegnazioni fornite dalla ricerca predefinita sono buone, quindi, optando per la decisione più vantaggiosa all'inizio della ricerca, si diminuiscono le possibilità di insuccesso e si velocizza il percorso verso la risoluzione.

L’euristica domWdeg con random value presenta il risultato migliore per il caso qc30\_05.

L’euristica domWdeg – random value + restarting presenta dei migliori risultati per diversi data files.

Nella maggior parte dei casi, metodo domWdeg + riavvio migliora le prestazioni perché il solutore dispone di alcune informazioni precedenti che può sfruttare per trovare una soluzione più velocemente e con meno fallimenti.

Il metodo genera informazioni aggiuntive (ovvero i pesi) durante l’esecuzione; perciò, quando la ricerca viene ravviata dal nodo iniziale, il solutore (in base alle informazioni generate in precedenza) seleziona le variabili successive, migliorando la ricerca e cercando di minimizzare gli errori iniziali dell'albero. Per l'appunto il riavvio della ricerca consente al solutore di avere l'opportunità di prendere decisioni diverse e migliori.

In questo contesto, osserviamo che il riavvio non si dimostra vantaggioso per le istanze 05 e 08, presumibilmente a causa dell'ordinamento casuale delle variabili, che risulta già ottimale. Nel dettaglio, questa circostanza potrebbe spiegare il motivo per cui il risolutore, inizialmente, non dedica il tempo necessario per esplorare il primo ramo durante la fase di riavvio, perdendo così il beneficio derivante dall'ordine già ottimo delle variabili.

Probabilmente il motivo è che la ricerca predefinita e l'euristica domWdeg hanno già un buon ordinamento delle variabili per queste istanze, che può produrre una buona soluzione. Pertanto, se il riavvio dell'istanza modificasse il buon ordine delle variabili assegnate, la ricerca potrebbe cambiare completamente o peggiorare.

When are random decisions (not) useful? Why?

A secondo del contesto specifico l’introduzione di casualità può risultare vantaggioso o svantaggioso.

Nel **problema delle nQueens** l’aggiunta di casualità comporta dei benefici: assegnare casualmente una posizione a una regina sulla scacchiera risulta più vantaggioso rispetto a un posizionamento sistematico nell'angolo in alto a sinistra. Questo perché la disposizione casuale delle prime regine aumenta la probabilità che queste controllino un numero maggiore di caselle rispetto al posizionamento minimo (in alto a sinistra). Tale approccio, sebbene renda più complesso il posizionamento successivo delle regine, contribuisce a ridurre lo spazio di ricerca in caso di fallimento.

Le decisioni casuali non sono utili nel problema del **posizionamento dei poster**, poiché le variabili casuali non migliorano i risultati del modello. Il posizionamento casuale di un poster all'interno della griglia aumenta la probabilità di insuccesso nei primi passaggi, soprattutto quando si tratta di un poster di grandi dimensioni.

Infine, nel **problema del quasi-gruppo**, l'introduzione della randomizzazione si dimostra utile, specialmente se combinata con il riavvio della ricerca.

Are dynamic heuristics always better than static ones? Why?

Le euristiche dinamiche non costituiscono necessariamente un miglioramento rispetto a quelle statiche. Al contrario, abbiamo osservato che in alcune circostanze, riordinando le variabili di istanza con euristiche specifiche del problema e risolvendo successivamente con euristiche statiche, si ottengono risultati più soddisfacenti. Nonostante ciò, quando non sono disponibili euristiche specifiche per il problema in esame, le euristiche dinamiche dimostrano di essere più utili in quanto possiedono il vantaggio di adattarsi allo stato attuale dell'albero di ricerca, potendo così reagire in modo più flessibile alle sfide specifiche incontrate durante il processo di risoluzione. Tuttavia, è importante notare che l'efficacia di questa adattabilità dipende dal problema e dalla natura della soluzione. In alcuni casi, le euristiche statiche possono essere sufficienti e più efficienti, soprattutto se le caratteristiche del problema rimangono costanti.

Is scheduling the search and/or restart always a good idea? Why?

La programmazione della **ricerca** e il **riavvio** sono strategie utilizzate per **migliorare l'efficacia della ricerca**. Il **riavvio** può essere utile per evitare di rimanere intrappolati in una parte poco promettente dello spazio di ricerca. Quando sono presenti sia la **randomizzazione** che i **riavvii**, possono eliminare l'enorme varianza nelle prestazioni del risolutore. Tuttavia, l'efficacia di queste strategie dipende fortemente dalla **configurazione specifica del problema** e **dell'algoritmo**. L'uso appropriato delle strategie di riavvio richiede una valutazione dettagliata e una sperimentazione nel contesto del problema affrontato.

La **sezione sul RIAVVIO** fornisce indicazioni sulla possibilità di riavviare la ricerca per correggere errori iniziali che potrebbero causare un'elevata quantità di ricerche inutili. Il riavvio può essere benefico per evitare di rimanere intrappolati in regioni non produttive dell'albero di ricerca. Tuttavia, si nota anche che il successo del riavvio sipende dalla strategia di ricerca sottostante e dalla presenza di casualità per introdurre variazioni significative.

Il problema si presenta quando vengono prese delle decisioni sbagliate all’inizio dell’albero e questo provoca una quantità esponenziale di ricerche per essere annullate. A tale problema esiste un metodo risolutivo ovvero: riavviare la ricerca dall’inizio per prendere decisioni diverse. In Minizinc esistono diverse strategie di riavvio: riavvio costante, riavvio geometrico e riavvio di Luby.

È importante notare che la ricerca con riavvio risulta efficace solo se la strategia di ricerca di base apporta variazioni significative durante ciascun nuovo avvio del processo. Per tale motivo è necessario utilizzare la randomizzazione (sia nella scelta delle variabili sia nella scelta dei valori). In alternativa, alcune strategie di selezione delle variabili fanno uso di informazioni raccolte da ricerche precedenti e quindi daranno luogo a comportamenti diversi, ad esempio dom\_w\_deg.

Nel problema del completamento di un quasigruppo, si nota che i risultati della ricerca con riavvio possono talvolta essere peggiori di quelli della ricerca predefinita. In particolare, il riavvio non sembra essere sempre una buona idea per alcune istanze, ciò può essere attribuito al fatto che la ricerca predefinita e l'euristica domWdeg presentano già un ordinamento efficace delle variabili in alcune istanze, che potrebbe portare a soluzioni migliori senza la necessità di riavviare il processo di ricerca.

IN QUESTA DOMANDA NON SO SE CANCELLARE QUELLO CHE HO SCRITTO SULLA RANDOMIZZAZIONE SICCOME NON è CITATA NELLA DOMANDA

You have done much better this time. Did you write all the text yourselves (in which case your English seems quite advanced) or did you use a tool to support your writing?

PP

The following sentences still do not explain why random decisions do not work well in PP. Note that there is nothing like random variable. The assigned values are chosen randomly. Can you try again to explain this?

"In the context of this problem, the application of randomness never leads to improvements; on the contrary, it increases the number of failures. The position of the posters is strongly influenced by the stringent constraints of the problem. Consequently, using a deterministic approach, which follows the constraints, produces better results than randomisation."

"Random decisions are not useful in the poster placement problem, as random variables do not improve the results of the model. The random placement of a poster within the grid increases the probability of failure in the first steps, especially when it is a large poster."

QCP

Which search method is the most robust overall?

Some corrections in terminology:

* Each variable initially includes all constraints -> each variable is included in all the constraints
* Error -> failure
* Search/Research scheduling -> programming search (research is a scientific investigation)

“You have done much better this time. Did you write all the text yourselves (in which case your English seems quite advanced) or did you use a tool to support your writing? “

Thank you for your feedback. Yes, we write the texts ourselves but we confess that, despite our best efforts, we often find it difficult to write in English. Using a support tool helps us to translate concepts understood in our mother tongue into English. We would like to emphasise that, despite these language difficulties, we are working hard to achieve results in this subject, in fact we have asked you to make a call just to prove it to us. We know that English is important and we are trying to improve in this as well.

Thank you also for the correction of terminology. We have corrected them in our report.  
  
“PP  
The following sentences still do not explain why random decisions do not work well in PP. Note that there is nothing like random variable. The assigned values are chosen randomly. Can you try again to explain this?

"In the context of this problem, the application of randomness never leads to improvements; on the contrary, it increases the number of failures. The position of the posters is strongly influenced by the stringent constraints of the problem. Consequently, using a deterministic approach, which follows the constraints, produces better results than randomisation."

"Random decisions are not useful in the poster placement problem, as random variables do not improve the results of the model. The random placement of a poster within the grid increases the probability of failure in the first steps, especially when it is a large poster."”

Let us try again to explain why random decisions do not work well in Poster Placement (PP).  
Variables are not chosen randomly in the context of the problem, but the values assigned to these variables are chosen randomly. In this context, randomly assigning values to variables does not improve the results of the model (i.e. does not lead to improvements).

The random placement of a poster in the grid increases the probability of failure in the first steps, especially when it is a large poster. In particular, the placement of a larger rectangle is more complex, as the associated constraints are more stringent than for smaller rectangles.

This variation in the constraints and size of the rectangles contributes to a significant difference. Rectangles that are more difficult to place tend to generate more frequent failures due to the greater rigidity of their constraints.

For these reasons, in the context of Poster Placement, it is crucial to follow a deterministic approach that respects constraints, as this tends to produce better results. By prioritising the search for posters that violate restrictions instead of starting with a random placement, we can minimise the risk of violating restrictions later in the resolution process, thus contributing to a more efficient and accurate solution.

“QCP  
Which search method is the most robust overall? Qual è il metodo di ricerca più robusto in assoluto?”  
The most robust search overall seems to be the domWdeg - random value + restarting method because it has better results for different data files.

In most cases, this method improves performance because the solver has some prior information that it can exploit to find a solution faster and with fewer failures.

The method generates additional information (e.g., weights) during execution; therefore, when the search is restarted from the initial node, the solver (based on the previously generated information) selects the next variables, improving the search and trying to minimize initial tree failures. In fact, restarting the search allows the solver to have the opportunity to make different and better decisions.

It is noted, however, that restarting does not prove beneficial for instances 05 and 08, presumably because of the random ordering of the variables, which is already optimal.

Yes I can see that you are working hard, that's what matters the most. Congratulations! And there's nothing wrong in using a support tool (I do that sometimes in Italian  I think seeing the translated text helps improve the language.  
  
Thanks for the answers, which I find satisfactory. One final remark: in QCP the most robust method is restarting because as you can see it always finds a solution (while the others sometimes cannot).