

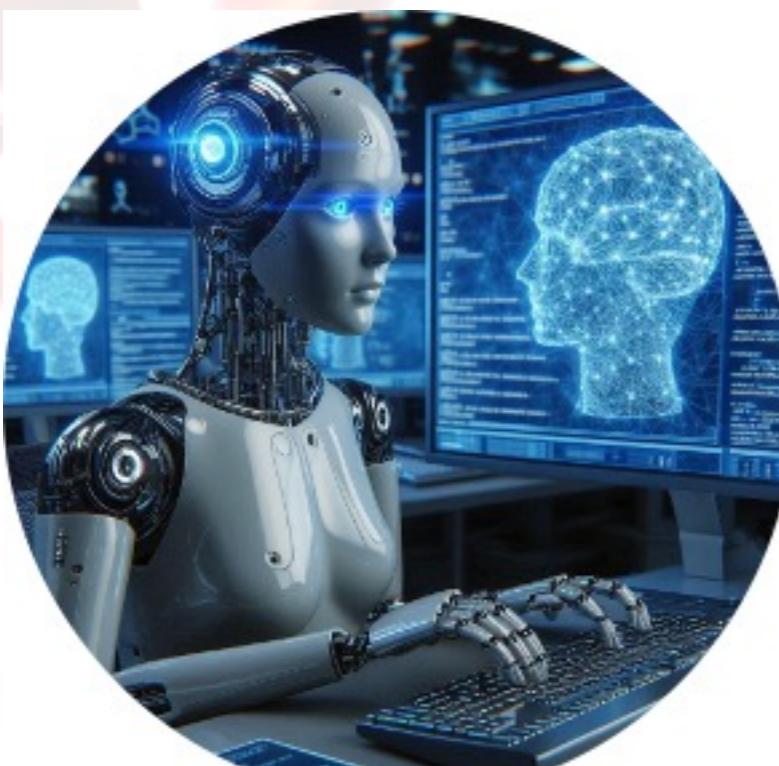
# Heuristics & Metaheuristics for Optimization & Learning



**Mario Pavone**

mpavone@dmi.unict.it

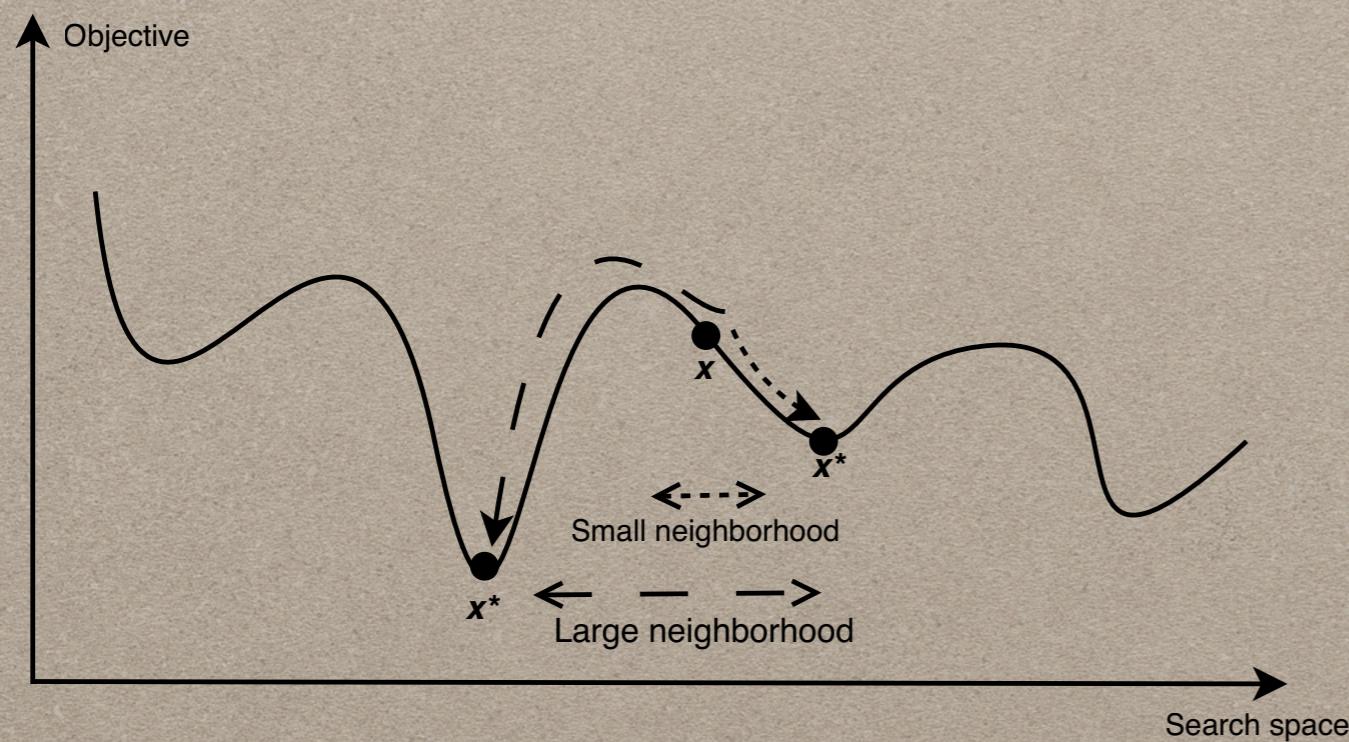
<http://www.dmi.unict.it/mpavone/>



# VERY LARGE NEIGHBORHOOD

**Compromise:** between size & quality of the neighborhood and computational complexity

Designing large neighborhood may improve the quality of the obtained solutions



**FIGURE 2.10** Impact of the size of the neighborhood in local search. Large neighborhoods improving the quality of the search with an expense of a higher computational time.

**Disadvantage:** additional computational time!!

The complexity of the search will be much higher.

Design efficient procedures to explore large neighborhoods

The size of the neighborhood for a solution  $s$  is the number of neighboring solutions of  $s$

# TABU SEARCH

TS behaves like a steepest LS algorithm, but it accepts nonimproving solutions to escape from local optima when all neighbors are nonimproving solutions. Usually, the

When a local optima is reached, the search carries on by selecting a candidate worse than the current solution

The best solution in the neighborhood is selected as the new current solution even if it is not improving the current solution.

To avoid cycles, TS discards the neighbors that have been previously visited. It memorizes the recent search trajectory. Tabu search manages a memory of the solutions or moves recently applied, which is called the *tabu list*. This tabu list constitutes the short-term memory. At each iteration of TS, the short-term memory is updated.

- **Tabu list:** The goal of using the short-term memory is to prevent the search from revisiting previously visited solutions. As mentioned, storing the list of all visited solutions is not practical for efficiency issues.

# TABU SEARCH

TS behaves like a steepest LS algorithm, but it accepts nonimproving moves to escape from local optima when all neighbors are nonimproving.

When a local minimum is reached, a new solution is selected by

The whole neighborhood is explored in a deterministic manner.

The current

the neighbors that have been previously visited. It prevents the search trajectory. Tabu search manages a memory of the solutions recently applied, which is called the *tabu list*. This tabu list constitutes short-term memory. At each iteration of TS, the short-term memory is updated.

- **Tabu list:** The goal of using the short-term memory is to prevent the search from revisiting previously visited solutions. As mentioned, storing the list of all visited solutions is not practical for efficiency issues.

# TABU SEARCH

---

**Algorithm 2.9** Template of tabu search algorithm.

```
s = s0 ; /* Initial solution */  
Initialize the tabu list, medium-term and long-term memories ;  
Repeat  
    Find best admissible neighbor s' ; /* non tabu or aspiration criterion holds */  
    s = s' ;  
    Update tabu list, aspiration conditions, medium and long term memories ;  
    If intensification_criterion holds Then intensification ;  
    If diversification_criterion holds Then diversification ;  
Until Stopping criteria satisfied  
Output: Best solution found.
```

---

# TABU SEARCH

Main design issues specific to a simple TS:

- **Tabu list:** The goal of using the short-term memory is to prevent the search from revisiting previously visited solutions. As mentioned, storing the list of all visited solutions is not practical for efficiency issues.
- **Aspiration criterion:** A commonly used aspiration criteria consists in selecting a tabu move if it generates a solution that is better than the best found solution. Another aspiration criterion may be a tabu move that yields a better solution among the set of solutions possessing a given attribute.

# TABU SEARCH

## Common Advanced Mechanisms in TS

- **Intensification (medium-term memory):** The medium-term memory stores the elite (e.g., best) solutions found during the search. The idea is to give priority to attributes of the set of elite solutions, usually in weighted probability manner. The search is biased by these attributes.
- **Diversification (long-term memory):** The long-term memory stores information on the visited solutions along the search. It explores the unvisited areas of the solution space. For instance, it will discourage the attributes of elite solutions in the generated solutions to diversify the search to other areas of the search space.

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

## Sudoku

- Il sudoku è un popolare rompicapo logico con i numeri. Le regole sono molto semplice. La griglia del sudoku è composta da  $N \times N$  caselle divisa in altre griglie di  $n \times n$  dette “regioni”. Dove  $n = \sqrt{N}$
- Per risolvere il sudoku devi seguire queste direttive:
  - Ogni riga deve contenere tutti i numeri da 1 a N.
  - Ogni colonna deve contenere tutti i numeri da 1 a N.
  - Ogni regione deve contenere tutti i numeri da 1 a N.

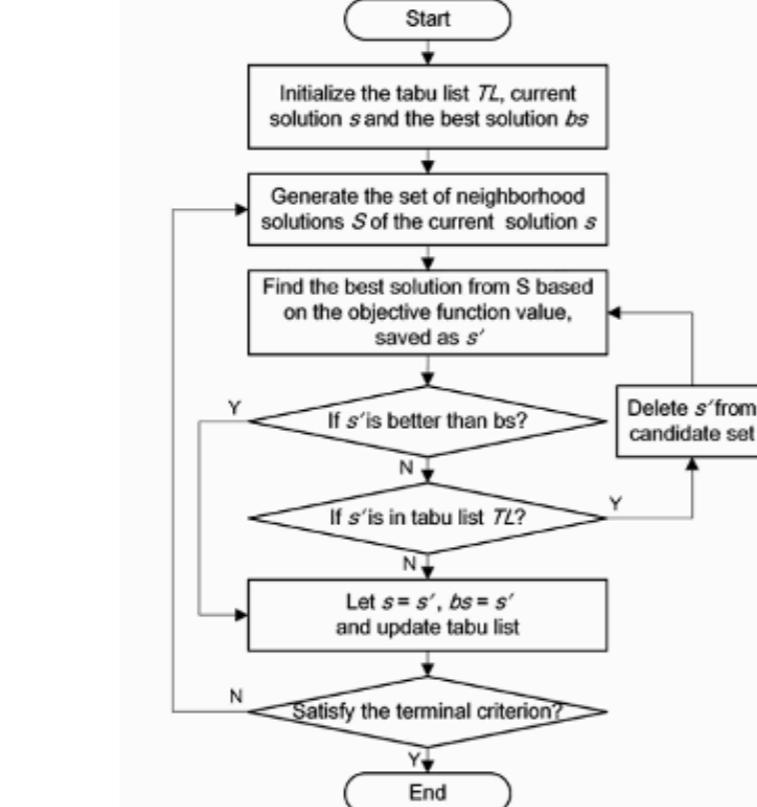
1								3
	7	2	6		4	8		
4		9	3	5			6	
	3	4	8		2			
	4	1	6	9	3			
	6			8	9			
5	7	8		4			2	
		3			7			
2							5	

Esempio di sudoku 9x9

23	20	8	15	21	6	17				24	19		12		2	
	14	3	11	18	2	13			17		15	18				
		4	7	14						1	8	3		11	22 24	
		6		23						19	10		25			
10	9		13	8	15	19	24		22	6		11	12	18	4	20
	3		10	25		16		4	21	9	22	8	20	12	7	17
20	25	8	1	18	19	24	9			22	14	2		3	11	
16	23	22			18	4	17		10	7	15	5			24	
21	5	19	12			11	8	20			9	16	15			18
	7	15		5	10	17	3			21	6	19	22	14		
18					23	2	10	5	20	3	21	1				
23	21	2	5	6	15	19	12	16			24	7	9	8		
	6	20			5	9	24	7		13	23	15	16	19		
				19	24	1	18	3	16	2	14	11	4	17	15	
	10			12	22	6	25	13	4	18		1		2		
2			5	23		22				9	25		21	1	11	
	21		18	11					13	2	5	17	8		25	16
	17	4			25		15	16	8	19		21	23	14	22	
					15	6		18	8	7	12		10			
18	22	14	13	24	6	20	3	7		11					12	
	1	7			8	13			20		5		17	22	16	9
	12	13	2	22	25	7			1	19		14	3	20	18	23
15	10		6	3	14	4			23	12	11	18	5	2	19	
	19	4	3	17	9	11	22		2	15			14	6	1	
14		24	23	1					6	9	25	22	4		12	

Esempio di sudoku 25x25

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE



## Recap e concetti chiave

- **Ricerca locale migliorativa:** Parte da una soluzione iniziale e cerca di migliorarla esplorando il vicinato.
- **Memoria Tabù:** Utilizza una lista tabù per evitare di tornare su soluzioni già visitate o per impedire cicli.
  - La lista tabù è una memoria a breve termine che registra le mosse o soluzioni recenti.
- **Accettazione di soluzioni subottimali:** Permette di superare minimi locali temporaneamente peggiorando la soluzione.
- **Strategie di diversificazione:** Introduce variazioni casuali o calcolate per esplorare nuove aree dello spazio delle soluzioni.

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

## Sudoku

- Il sudoku è un popolare rompicapo logico con i numeri. Le regole sono molto semplice. La griglia del sudoku è composta da  $N \times N$  caselle divisa in altre griglie di  $n \times n$  dette “regioni”. Dove  $n = \sqrt{N}$
- Per risolvere il sudoku devi seguire queste direttive:
  - Ogni riga deve contenere tutti i numeri da 1 a N.
  - Ogni colonna deve contenere tutti i numeri da 1 a N.
  - Ogni regione deve contenere tutti i numeri da 1 a N.

1								3
	7	2	6		4	8		
4		9	3	5			6	
	3	4	8		2			
	4	1	6	9	3			
	6			8	9			
5	7	8		4			2	
		3			7			
2							5	

Esempio di sudoku 9x9

23	20	8	15	21	6	17					24	19		12		2
	14	3	11	18	2	13			17		15	18				
		4	7	14						1	8	3		11	22	24
		6		23						19	10		25			
10	9		13	8	15	19	24		22	6		11	12	18	4	20
	3		10	25		16		4	21	9	22	8	20	12	7	17
20	25	8	1	18	19	24	9			22	14	2		3	11	
16	23	22			18	4	17		10	7	15	5			24	
21	5	19	12			11	8	20			9	16	15			18
	7	15		5	10	17	3			21	6	19	22	14		
18					23	2	10	5	20	3	21	1				
23	21	2	5	6	15	19	12	16			24	7	9	8		
	6	20			5	9	24	7		13	23	15	16	19		
				19	24	1	18	3	16	2	14	11	4	17	15	
	10			12	22	6	25	13	4	18		1		2		
2		5	23		22					9	25		21	1	11	
	21		18	11					13	2	5	17	8		25	16
	17	4			25		15	16	8	19		21	23	14	22	
					15	6		18	8	7	12		10			
18	22	14	13	24	6	20	3	7		11					12	
	1	7			8	13			20		5		17	22	16	9
	12	13	2	22	25	7			1	19		14	3	20	18	23
15	10		6	3	14	4			23		12	11	18	5	2	19
	19	4	3	17	9	11	22		2	15			14	6	1	
14		24	23	1					6	9	25	22	4		12	

Esempio di sudoku 25x25

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

23	20	8	15	21	6	17			24	19		12		2		
	14	3	11	16	2	13		17		15	18					
	4	7	14				1		8	3		11	22	24		
	6	23					19	10		25						
10	9	13	8	15	19	24		22	6	11	12	18	4	20		
	3	10	25		16		4	21	9	22	8	20	12	7	17	
20	25	8	1	18	19	24	9		22	14	2		3	11		
16	23	22		18	4	17		10	7	15	5		24			
21	5	19	12			11	8	20	1	9	16	15		18		
	7	15		5	10	17	3		21	6	19	22	14			
18				23	2	10	5	20	3	21	1					
23	21	2	5	6	15	19	12	16		24	7	9	8			
	6	20			5	9	24	7		13	23	15	16	19		
				19	24	1	18	3	16	2	14	11	4	17	15	
	10			12	22		6	25	13	4	18		1		2	
2		5	23		22				9	25		21	1		11	
21		18	11					13	2	5	17	6		25	16	
	17	4			25	15	16	8	19		21	23	14	22		
				15	6		18		8	7	12		10			
18	22	14	13	24	6	20	3	7		11				12		
	1	7			8	13		20		5		17	22	16	9	
12	13	2	22	25	7			1	19		14	3	20	18	23	
15	10		6	3	14	4		23		12	11	18	5	2	19	
	19	4		3	17	9	11	22		2	15		14	6	1	
14			24	23	1			6		9	25	22		4		12

1					3
	7	2	6	4	8
4		9	3	5	6
	3	4	8	2	
	4	1	6	9	3
	6		8	9	
5	7	8	4		2
	3		7		
2					5

*Cosa ci serve per il TS per risolvere il sudoku?*

- Soluzione Iniziale
- Funzione di Fitness
- Funzione di generazione del vicinato
- Come utilizzare la Memoria Tabù

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

5	2	4	4	3	7	6	8	9
6	7	9	1	5	9	2	7	3
8	1	3	6	8	2	4	1	5
4	3	1	6	7	5	4	8	1
5	8	6	1	8	9	9	3	2
7	9	2	2	3	4	7	6	5
2	8	9	7	1	5	8	4	2
6	4	5	2	9	3	9	1	3
3	1	7	8	6	4	7	5	6

## *Generazione della soluzione iniziale*

- Un possibile approccio per generare una soluzione iniziale consiste nel leggere tutti i valori fissi, cioè gli elementi "fixed" del problema, ossia i numeri già presenti nella griglia.
- Per ciascun sottoblocco, si distribuiscono casualmente gli elementi rimanenti utilizzando i valori disponibili del sottoblocco, senza considerare eventuali violazioni delle regole relative a righe e colonne.
- Nell'immagine, i valori in verde rappresentano gli elementi fixed, mentre i valori in bianco indicano quelli inseriti dall'utente.
- In questo modo si avranno duplicati solo lungo le righe e le colonne!

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

1	5	2	4	4	3	7	6	8	9
2	6	9	7	1	5	9	2	7	3
2	8	1	3	6	8	2	4	1	5
2	4	3	1	6	7	5	4	8	1
2	5	8	6	1	8	9	9	3	2
2	7	9	2	2	3	4	7	6	5
2	2	8	9	7	1	5	8	4	2
2	6	4	5	2	9	3	9	1	3
2	3	1	7	8	6	4	7	5	6

2 3 1 3 2 3 3 2 3

## Fitness function

- Come funzione di fitness, possiamo sfruttare il fatto che nei sottoblocchi non possono esserci duplicati e concentrare il calcolo delle violazioni su righe e colonne. In particolare, si conta il numero totale di violazioni, ossia i duplicati presenti in ogni riga e in ogni colonna.
- La funzione di costo, quindi, è definita come la somma totale delle violazioni:
  - Somma delle violazioni per ogni riga.
  - Somma delle violazioni per ogni colonna.
- Nell'immagine mostrata, il costo della soluzione proposta è 38.
- Il problema dunque è di minimizzazione

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

## *Funzione di generazione del vicinato*

5	2	4
6	7	9
8	1	3

5	2	4
6	9	7
8	1	3

La generazione del vicinato avviene tramite la modifica controllata della griglia attuale, seguendo questi passaggi:

- Selezione casuale di un blocco:** Si sceglie in modo casuale uno dei sottoblocchi 3x3 della griglia.
- Identificazione degli elementi modificabili:** All'interno del blocco selezionato, si identificano i valori **non fissati** (ossia quelli non marcati come "fixed").
- Scambio casuale:** Si selezionano casualmente due valori **non fissati** all'interno del blocco e si scambiano tra loro. Questo processo genera una nuova configurazione della griglia, che rappresenta un vicino rispetto alla soluzione corrente.

# TABU SEARCH FOR SOLVING SUDOKU PUZZLE

*Utilizzo della memoria tabù (esempio)*

- **Memorizzazione delle mosse recenti:**
  - Ogni volta che si scambiano due valori **non fissati** in un blocco 3x3, la mossa viene registrata nella lista tabù.
- **Durata della tabù (lunghezza della lista):**
  - La memoria tabù ha una lunghezza limitata (ad esempio, 10 mosse). Quando viene aggiunta una nuova mossa e la lista è piena, la mossa più vecchia viene rimossa.
  - Questo garantisce che la memoria sia "a breve termine" e non limiti troppo la ricerca.
- **Controllo tabù:**
  - Prima di effettuare uno scambio, si verifica se la mossa è presente nella lista tabù.
  - Se è tabù, la mossa viene scartata, a meno che non soddisfi un **criterio di aspirazione** (vedi punto successivo).
- **Criterio di aspirazione:**
  - Anche una mossa tabù può essere accettata se porta a una soluzione con un costo migliore di qualunque soluzione trovata finora (criterio di aspirazione).
  - Questo evita che la memoria tabù ostacoli il progresso verso l'ottimo.
- **Aggiornamento della memoria:**
  - Dopo ogni iterazione, si aggiunge la mossa appena eseguita nella lista tabù per evitare di tornare subito alla configurazione precedente.