

Une distance entre deux ensembles de séquences avec contrainte de continuité

Application à des données en-ligne

Jinpeng Li, Harold Mouchère, Christian Viard-Gaudin

CIFED 2012

Plan

- 1: Définition du problème
- 2: DTW (Dynamic Time Warping)
 - entre deux symboles mono-traits
 - entre deux symboles multi-traits
- 4: Accélérer la recherche: A *
- 5: Étude expérimentale
- 6: Conclusion

Définition du problème

Reconnaissance de l'écriture

1: Segmentation des symboles,

2: Reconnaissance des symboles.

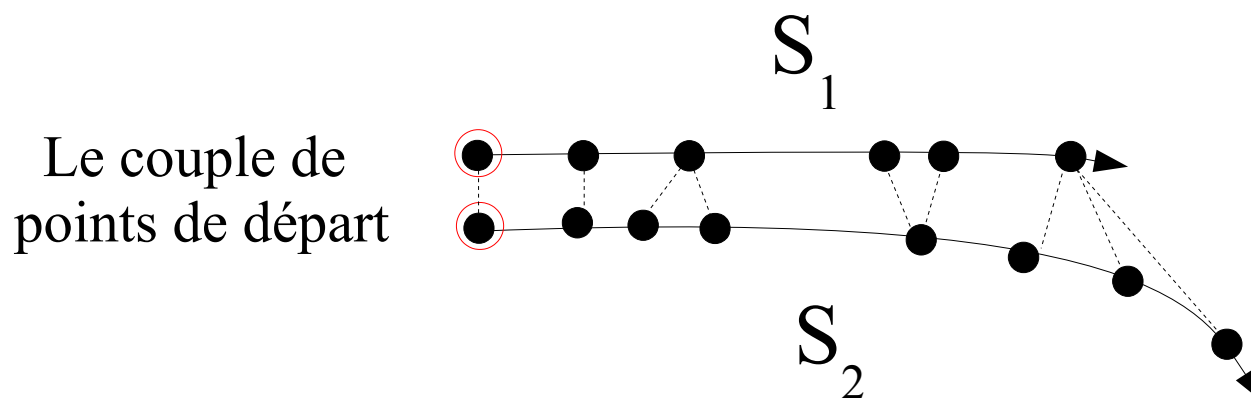
Les symboles isolés **mono-traits**: 

Deux symboles isolés **multi-traits**: 

[1] Chan, K.-F. & Yeung, D.-Y.
Mathematical Expression Recognition: A Survey 2000

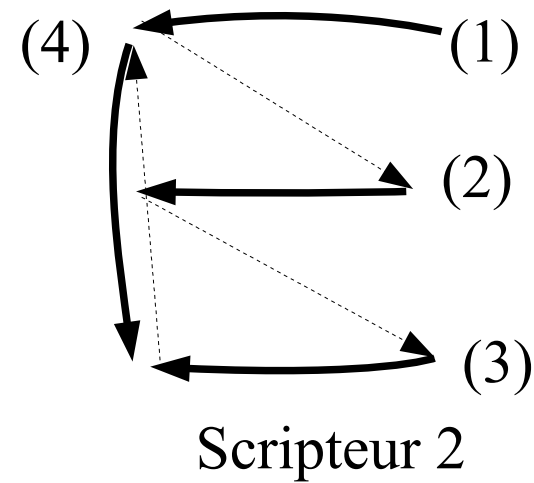
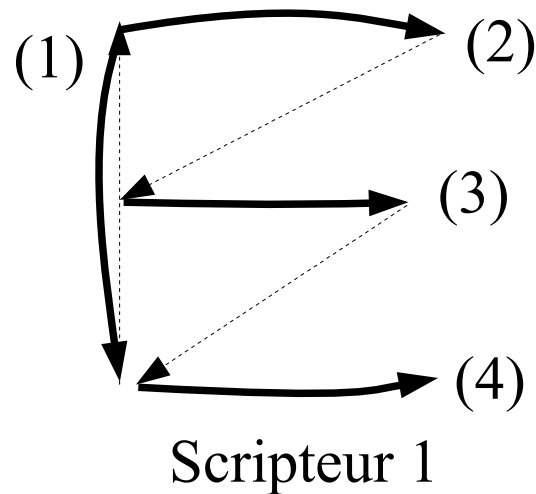
Alignement entre deux symboles mono-traits

L'algorithme **DTW** (Dynamic Time Warping) permet une mise en correspondance point à point en respectant la **contrainte de continuité** pendant l'alignement entre **les deux séquences**.



[1] Phdthesis (Vuori2002) Vuori, V.
Adaptive Methods for On-Line Recognition of Isolated Handwritten
Characters Helsinki University of Technology (Espoo, Finland), 2002

Alignement entre deux symboles multi-traits



Deux Exemples de Symbole E

On revient au cas de mono-trait!

(.) : Index de l'ordre de trait

→ : Sens de trait

1. Les traits sont concaténés en un trait.

2. Calcul d'une distance par DTW.

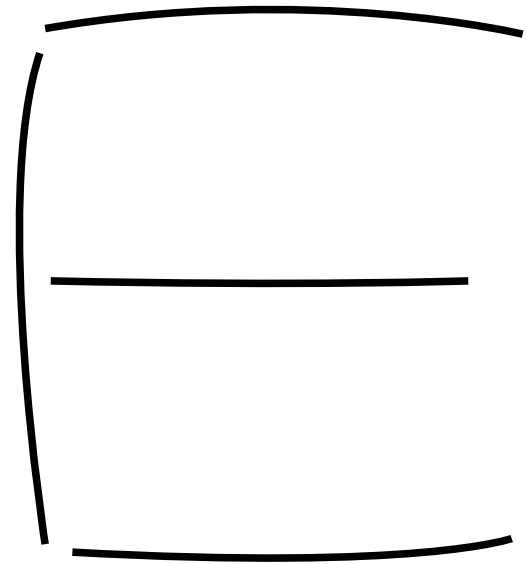
[1] Tan, G. X.; Viard-Gaudin, C. & Kot, A. C.
Automatic writer identification framework for online handwritten documents
using character prototypes Pattern Recogn., Elsevier Science Inc., 2009

Lire un Symbole



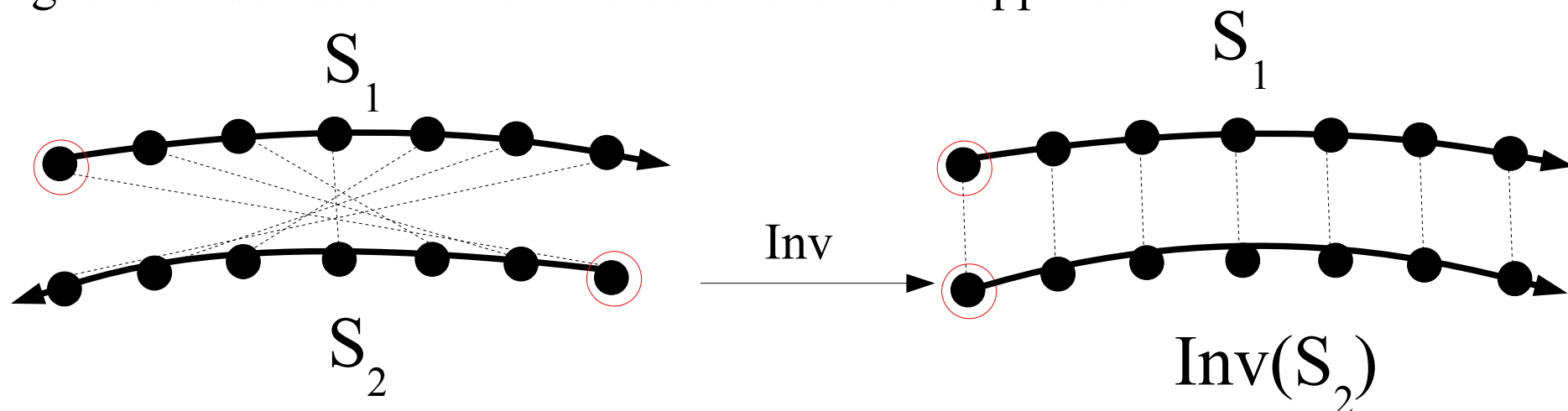
1: Pas d'information d'ordre de trait

2: Pas d'information de sens de trait

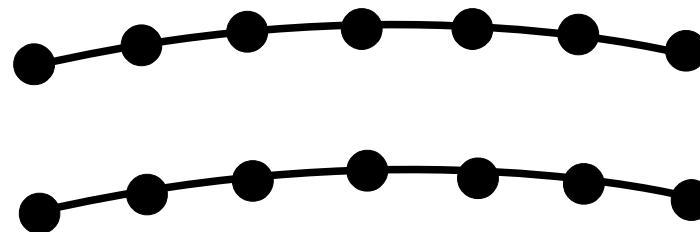


Une solution simple pour deux symboles mono-traits

L'alignement des deux traits avec deux directions opposées:



Très grande! $\leftarrow DTW(S_1, S_2) > DTW(S_1, inv(S_2))$
distance = $\min(DTW(S_1, S_2), DTW(S_1, inv(S_2)))$



Dans ce cas, la direction est ignorée

Alignement entre deux symboles multi-traits

Nombre de traits (N)	Exemple	Nombre de séquences (S)	Illustrations des tracés
1	—	2	→ ←
2	=	8	$\begin{matrix} (1) \rightarrow & \rightarrow & \leftarrow & \leftarrow & (2) \rightarrow & \rightarrow & \leftarrow & \leftarrow \\ (2) \rightarrow & \leftarrow & \rightarrow & \leftarrow & (1) \rightarrow & \leftarrow & \rightarrow & \leftarrow \end{matrix}$
3	≡	48	$\begin{matrix} (1) \rightarrow & (2) \rightarrow & & & & & & & (3) \leftarrow & (2) \leftarrow \\ \downarrow & \downarrow & \rightarrow & & & & & & \uparrow & \leftarrow \\ (3) \leftarrow & \leftarrow & & & & & & & (1) \rightarrow & \rightarrow \end{matrix}$
4	≡≡	384

$$S = N! \times 2^N \quad \text{Explosion du nombre de séquences !}$$

DTW

entre deux symboles mono-traits

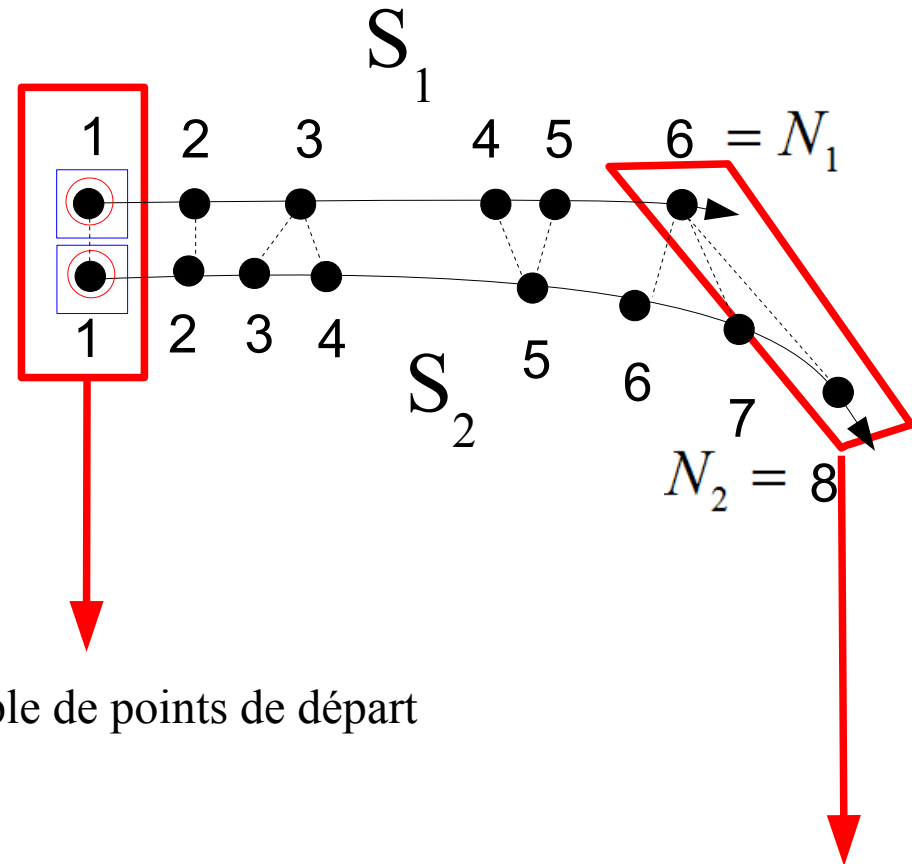
Définir un couple de départ et de fin

1: Nous définissons d'abord un couple de points de départ et un couple de points de fin.

$$S_1 = (p_1(1), \dots, p_1(N_1))$$

$$S_2 = (p_2(1), \dots, p_2(N_2))$$

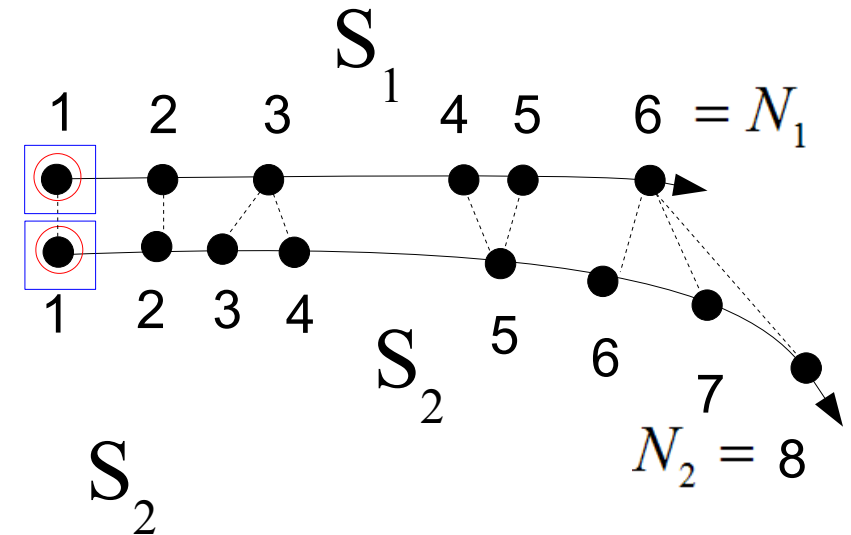
Le couple de points de départ



Le couple de points de fin

Chercher le meilleur chemin

1: Nous définissons d'abord un couple de points de départ et un couple de points de fin.



	1	2	3	4	5	6	7	8
1								14
2		2						13.3
3			3	4.3				13
4					4.5			12.2
5					5.5	6.9		11.2
6	9.5	9	8	7	6.8	6.7	8.2	10.2

Traçage en arrière

DTW minimise la somme de distances

1: Nous définissons d'abord un couple de points de départ et un couple de points de fin.

2: Obtenir le meilleur chemin.

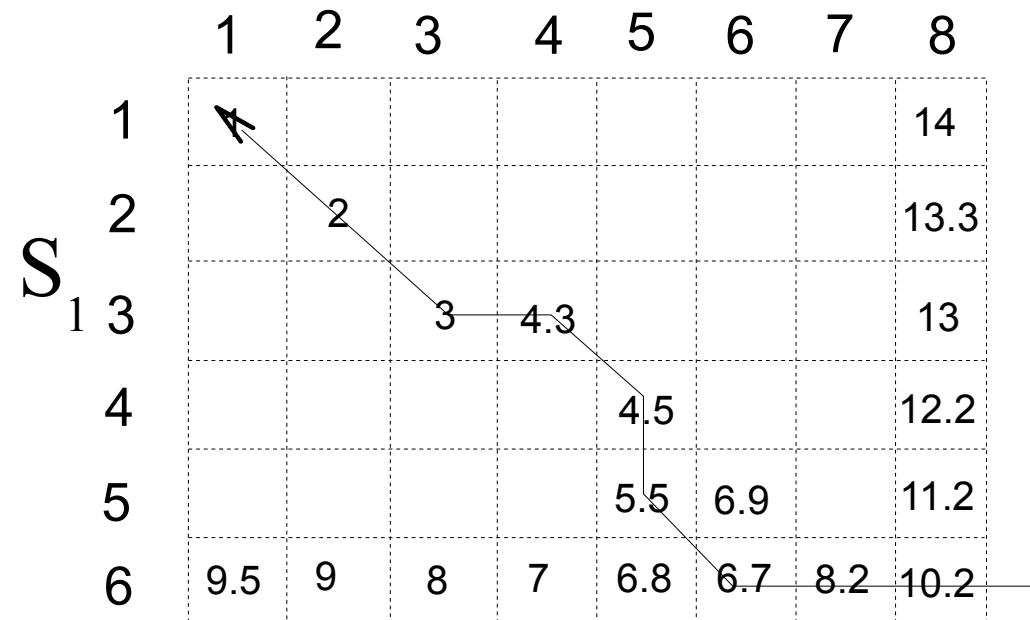
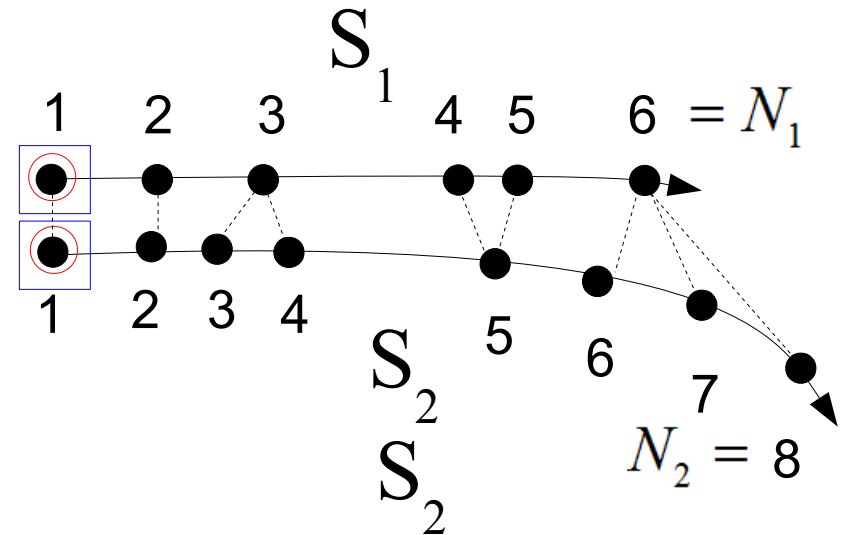
$$P(h) = (i(h), j(h)), 1 \leq h \leq H$$

$$P(1), \dots, P(8) =$$

$$(1,1), (2,2), (3,3), (4,4)$$

$$(5,5), (6,6), (7,6), (8,6)$$

$$D(S_1, S_2) = \min_{P(h)} \sum_{h=1}^H d(p_1(i(h)), p_2(j(h)))$$

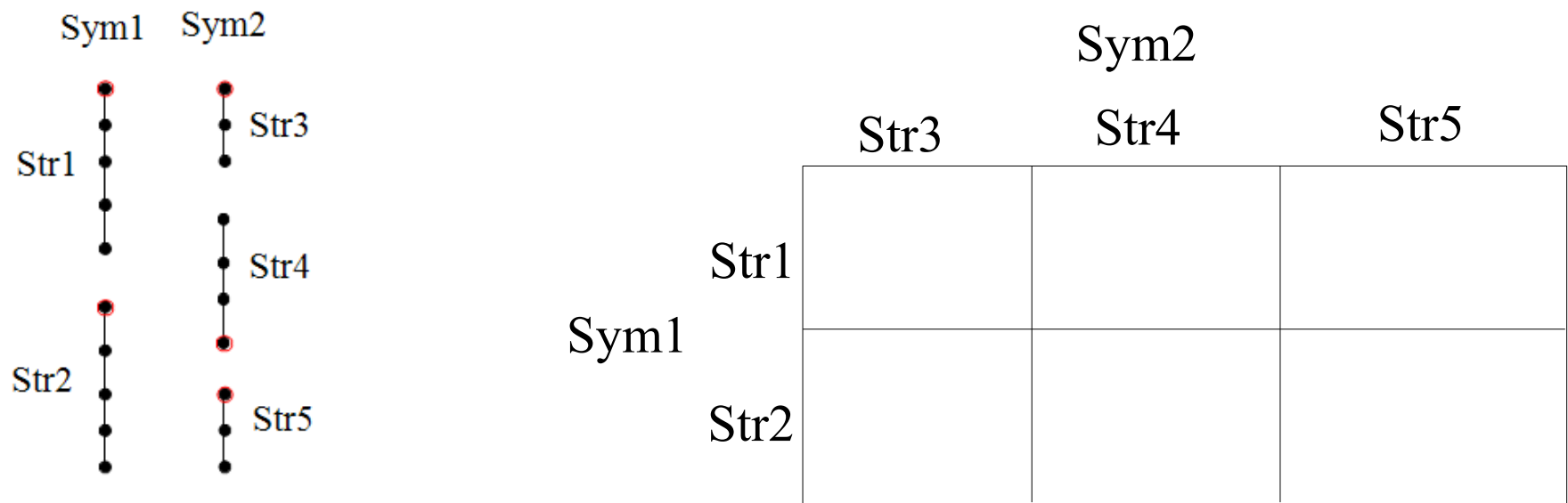


DTW


entre deux symboles multi-traits

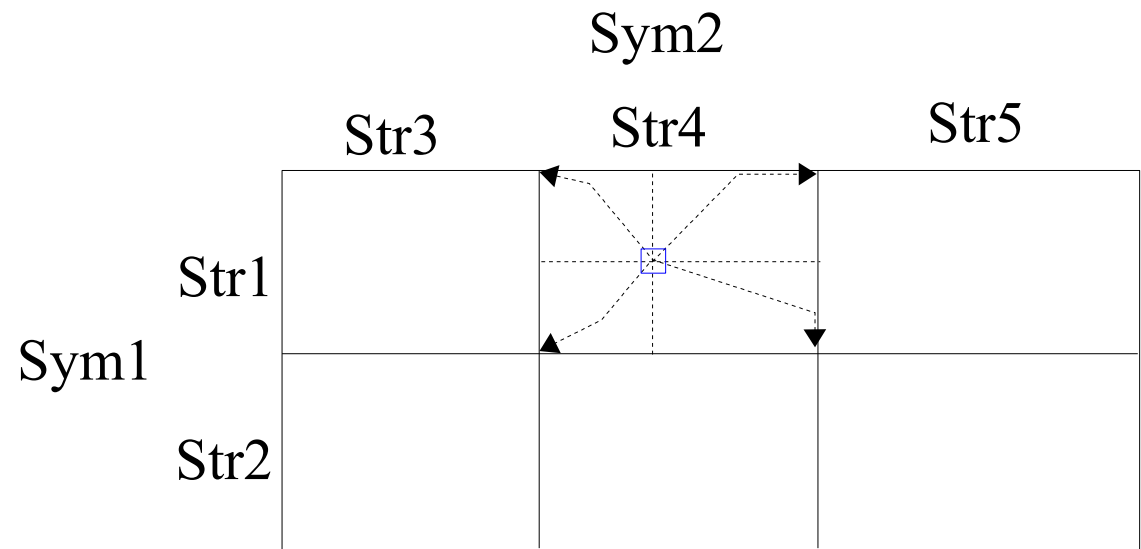
Une matrice pour deux symboles multi-traits

Soit **deux** symboles contenant respectivement **deux traits** et **trois traits**, les traits de ces deux symboles sont représentés respectivement en ligne et en colonne dans la matrice.



Quatre directions à partir d'un couple de départ

- 1: Nous définissons d'abord un couple de points de départ  .
- 2: Quatre directions d'alignement sont possibles.
- 3: Elles correspondent aux quatre orientations diagonales.
quatre matrices d'accumulation

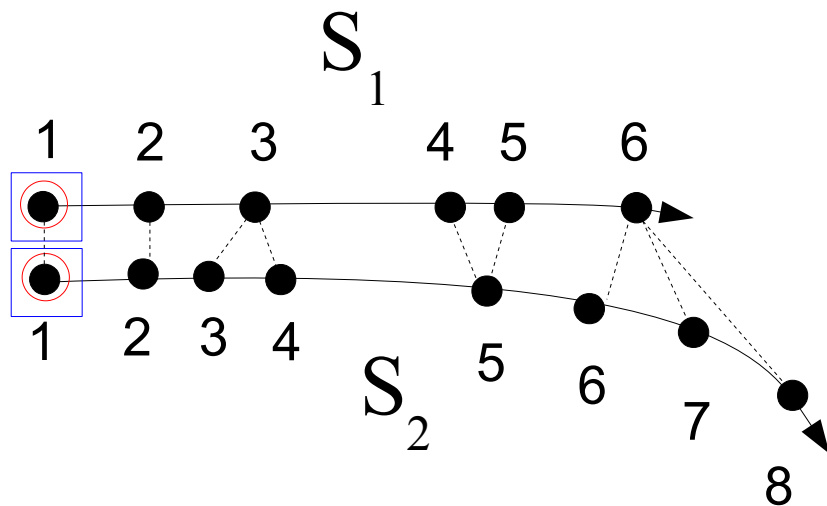


Nouveau couple de fin

1: Entrée: un couple de points de départ

2: Sortie: un couple de points de fin

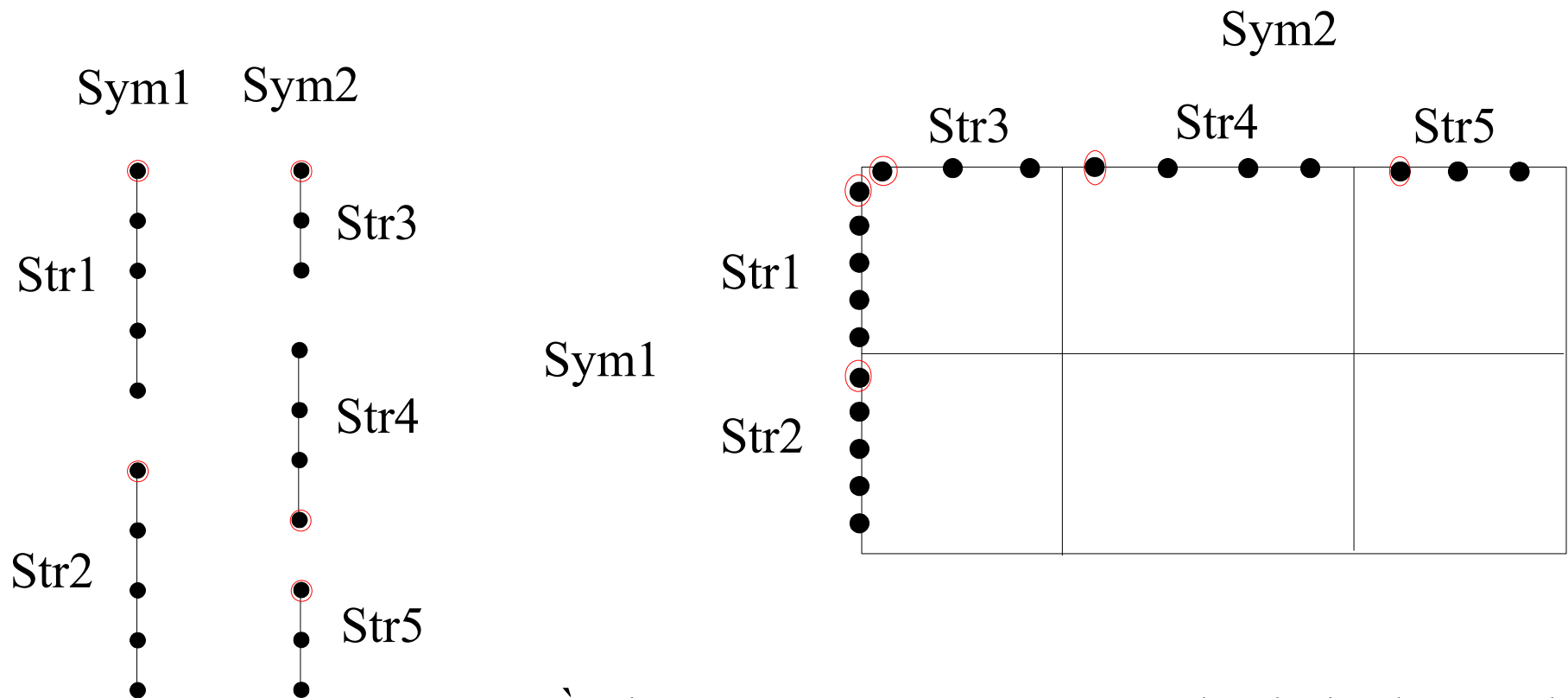
Il faut finir au moins un trait.



		S_2							
		1	2	3	4	5	6	7	8
S_1	1	1							14
	2		2						13.3
	3			3	4.3				13
	4					4.5			12.2
	5					5.5	6.9		11.2
	6	9.5	9	8	7	6.8	6.7	8.2	10.2

Choisir la valeur minimum sur les deux côtés.

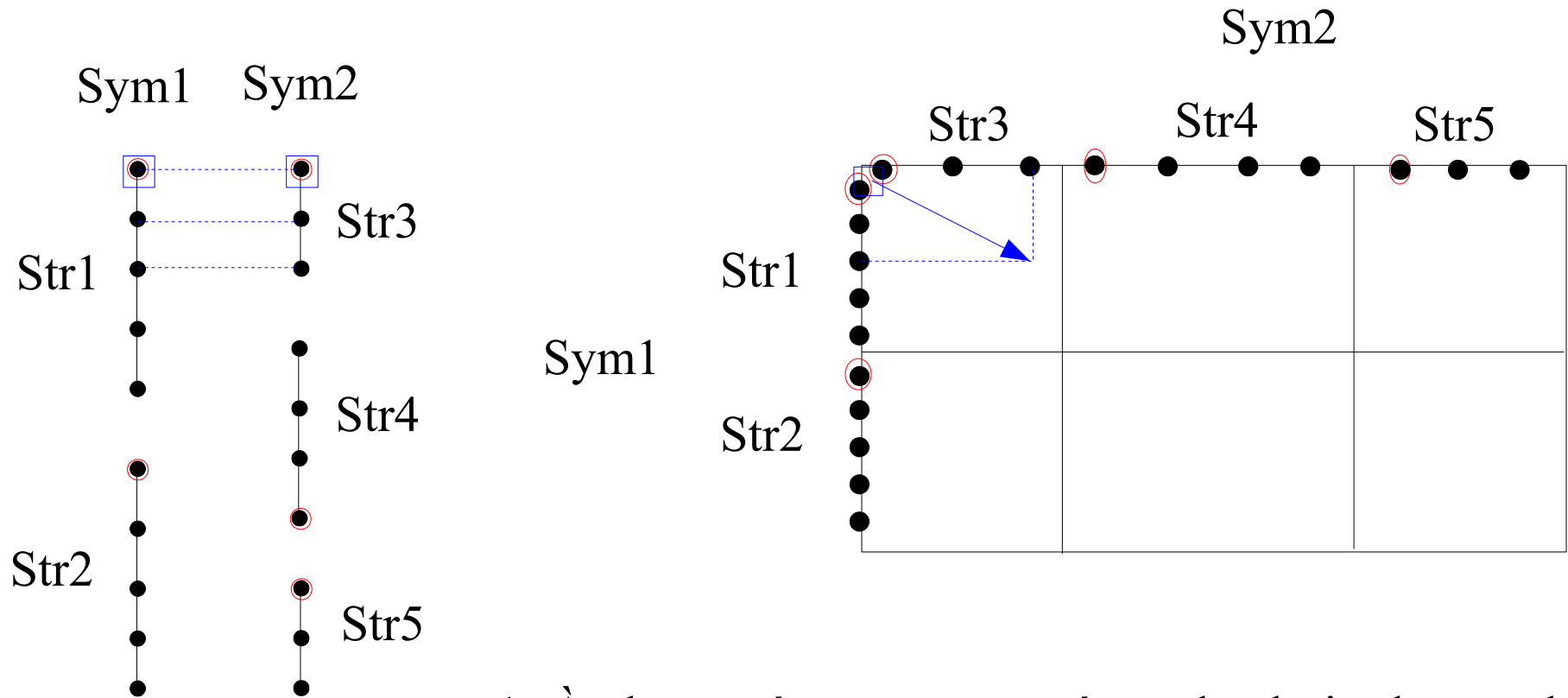
Un exemple d'alignement



1: À chaque étape nous répétons le choix de couple de points parmi les points non-utilisés.

2: La procédure sera finie lorsque tous les points seront utilisés au moins une fois.

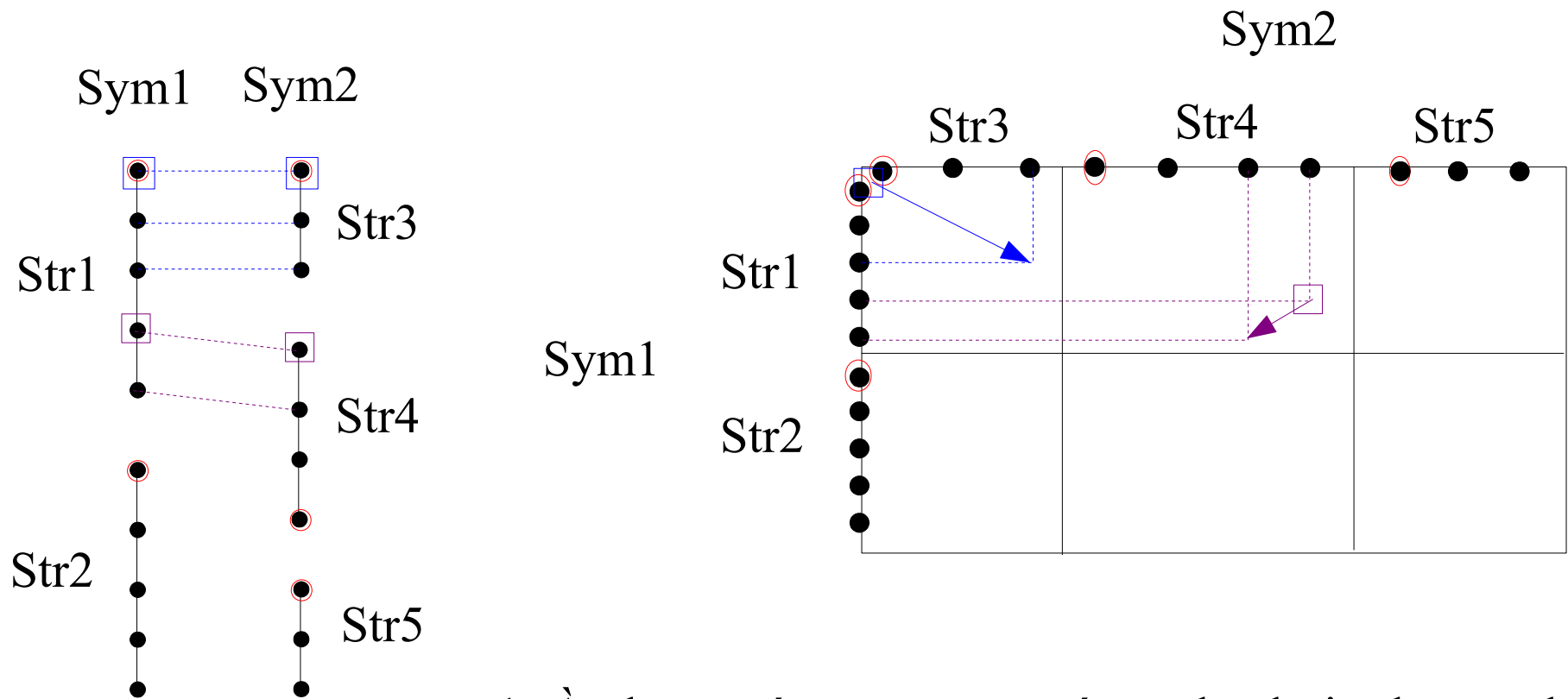
Un exemple d'alignement



1: À chaque étape nous répétons le choix de couple de points parmi les points non-utilisés.

2: La procédure sera finie lorsque tous les points seront utilisés au moins une fois.

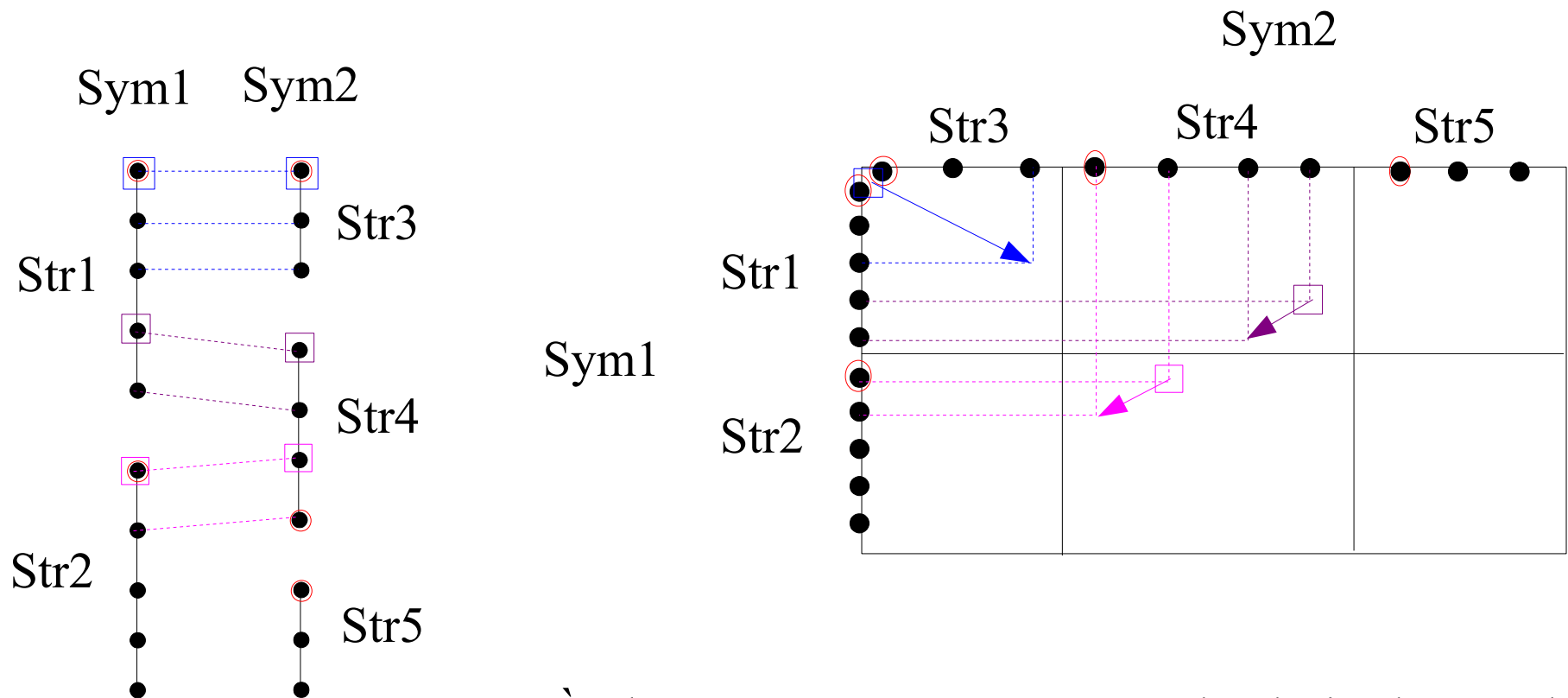
Un exemple d'alignement



1: À chaque étape nous repétons le choix de couple de points parmi les points non-utilisés.

2: La procédure sera finie lorsque tous les points seront utilisés au moins une fois.

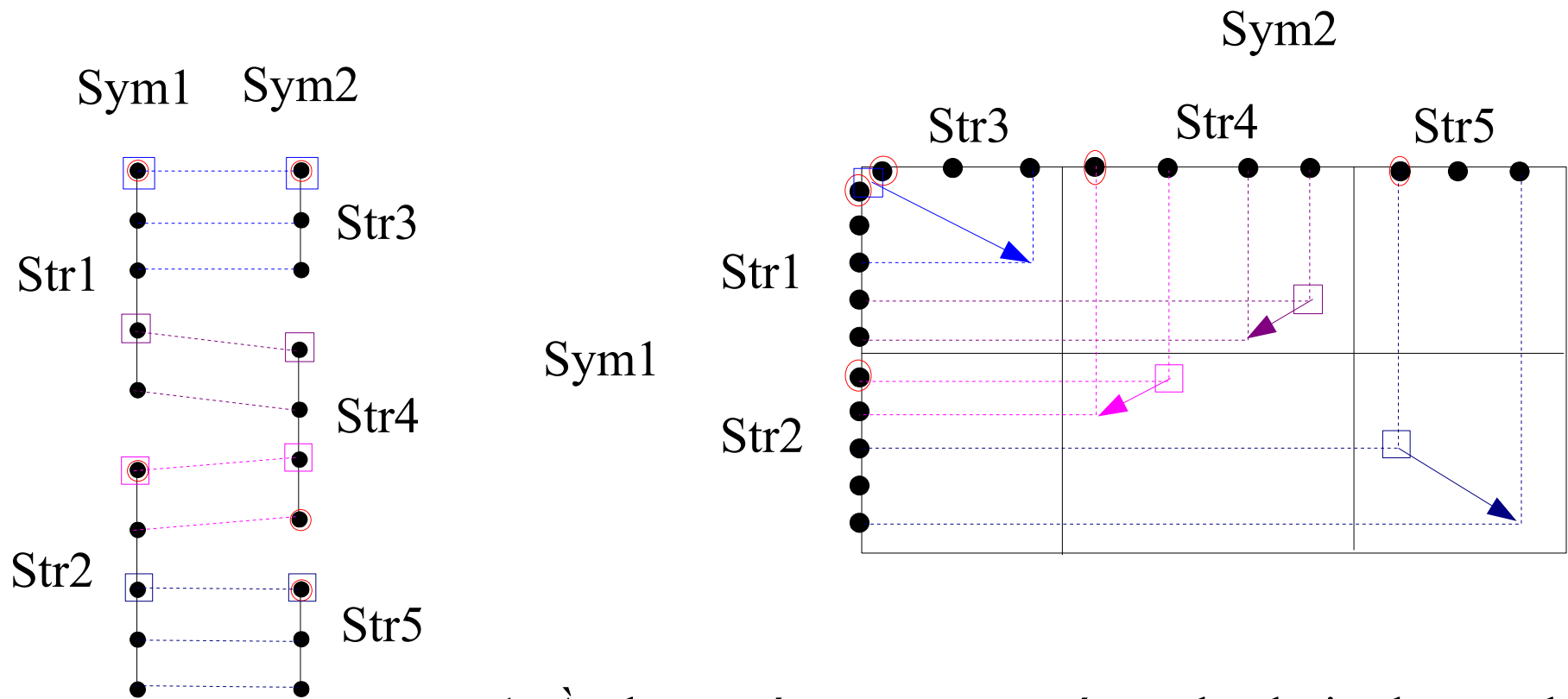
Un exemple d'alignement



1: À chaque étape nous répétons le choix de couple de points parmi les points non-utilisés.

2: La procédure sera finie lorsque tous les points seront utilisés au moins une fois.

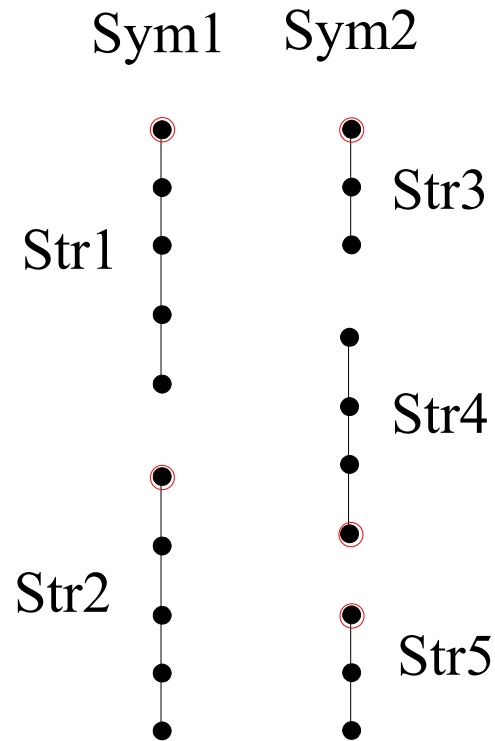
Un exemple d'alignement



1: À chaque étape nous répétons le choix de couple de points parmi les points non-utilisés.

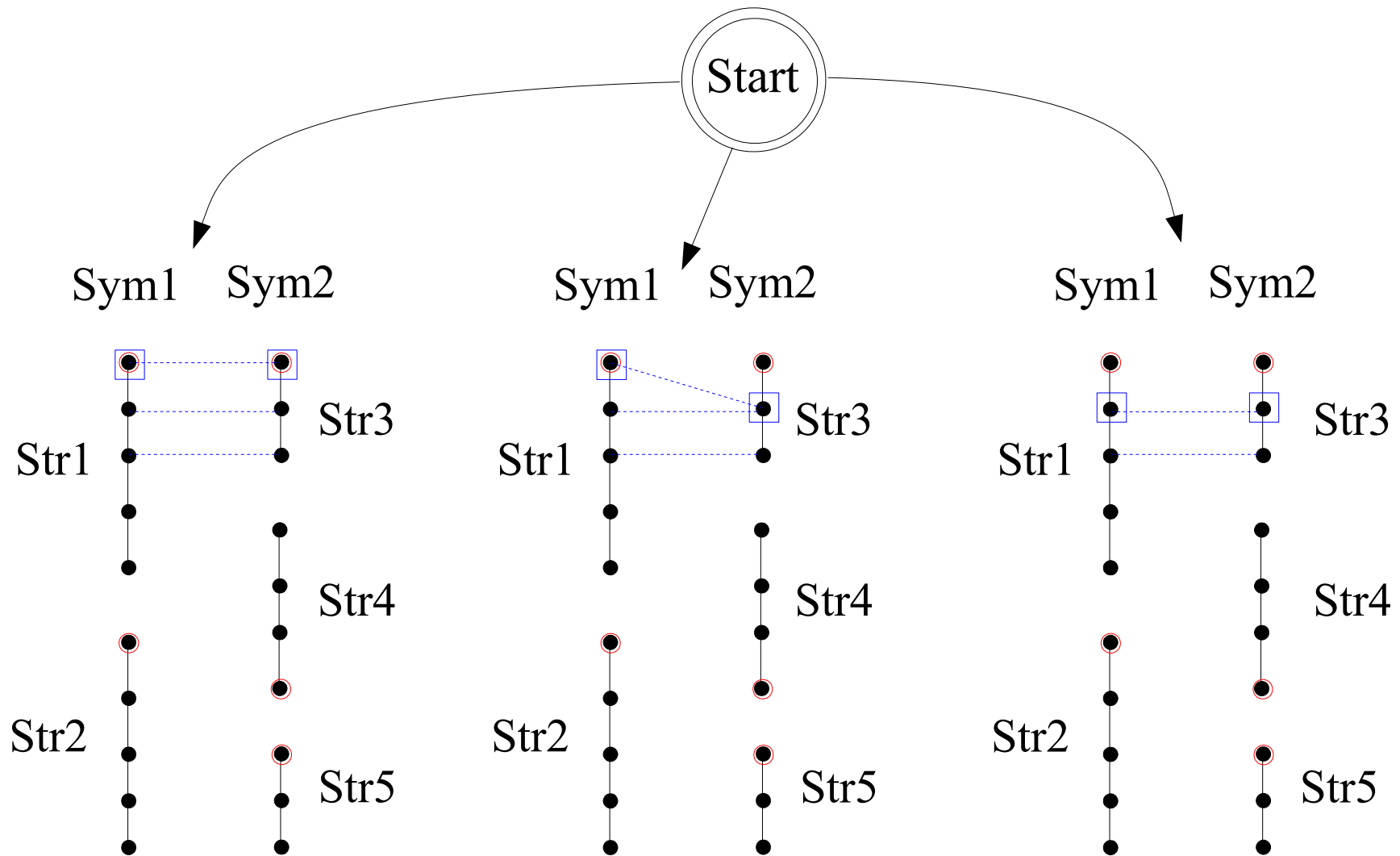
2: La procédure sera finie lorsque tous les points seront utilisés au moins une fois.

Recherche du meilleur chemin

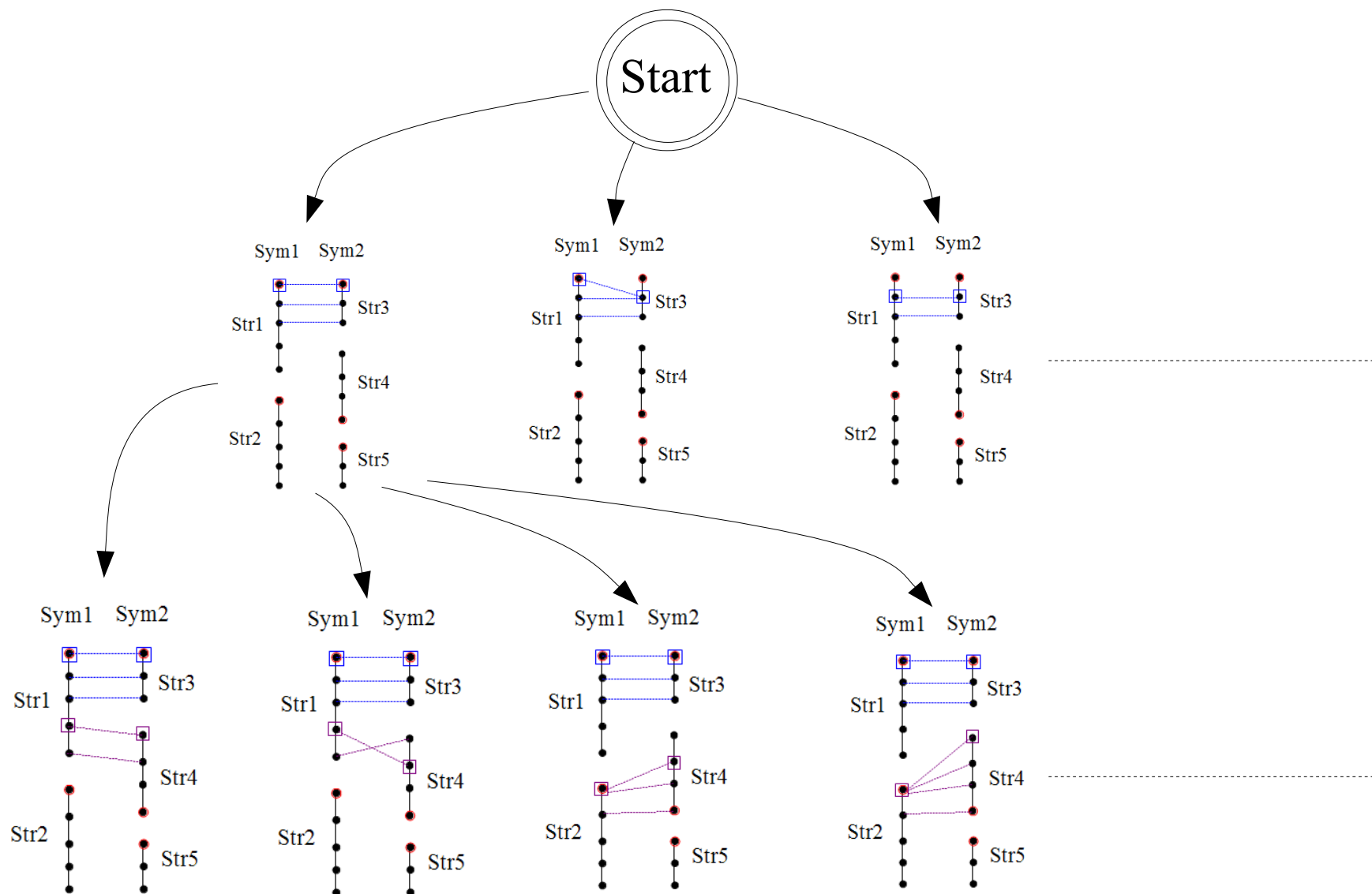


$$D(Sym1, Sym2) = \frac{1}{H} \min_{P(h)} \sum_{h=1}^H d(p_1(i(h)), p_2(j(h)))$$

Recherche du meilleur chemin



Recherche du meilleur chemin



A * (A étoile)

L'algorithme A*

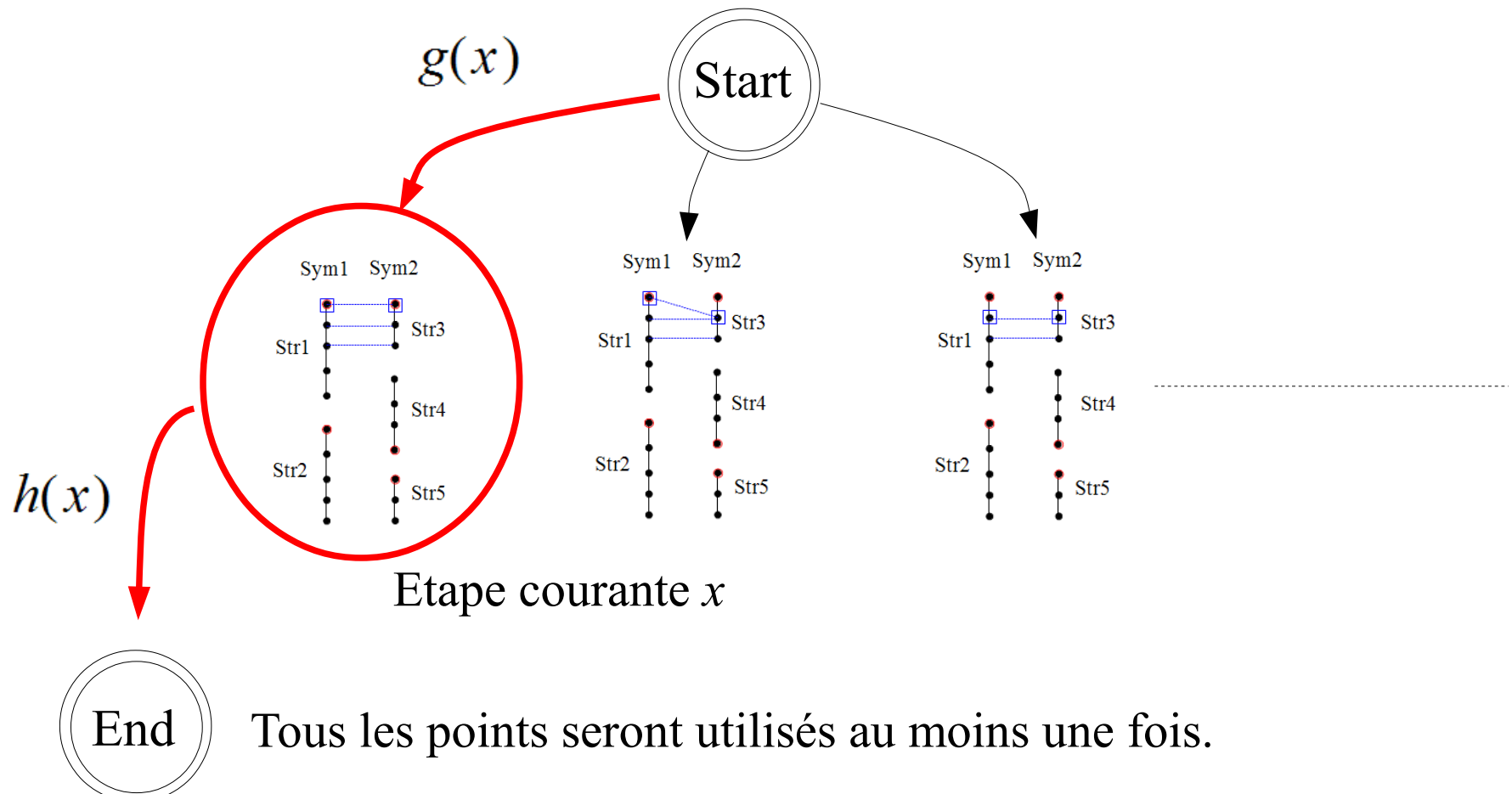
1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$

L'algorithme A*

1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$



L'algorithme A*

1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$

$$g(x) = \sum_{H_x} d(p_1(i(h)), p_2(j(h)))$$

$$h(x) = \frac{1}{2}(h_{sub}(x, Sym1, Sym2) + h_{sub}(x, Sym2, Sym1))$$

$h(x)$ correspond au coût nécessaire minimum pour aller de l'étape actuelle x à l'étape finale.

➤ C'est admissible.

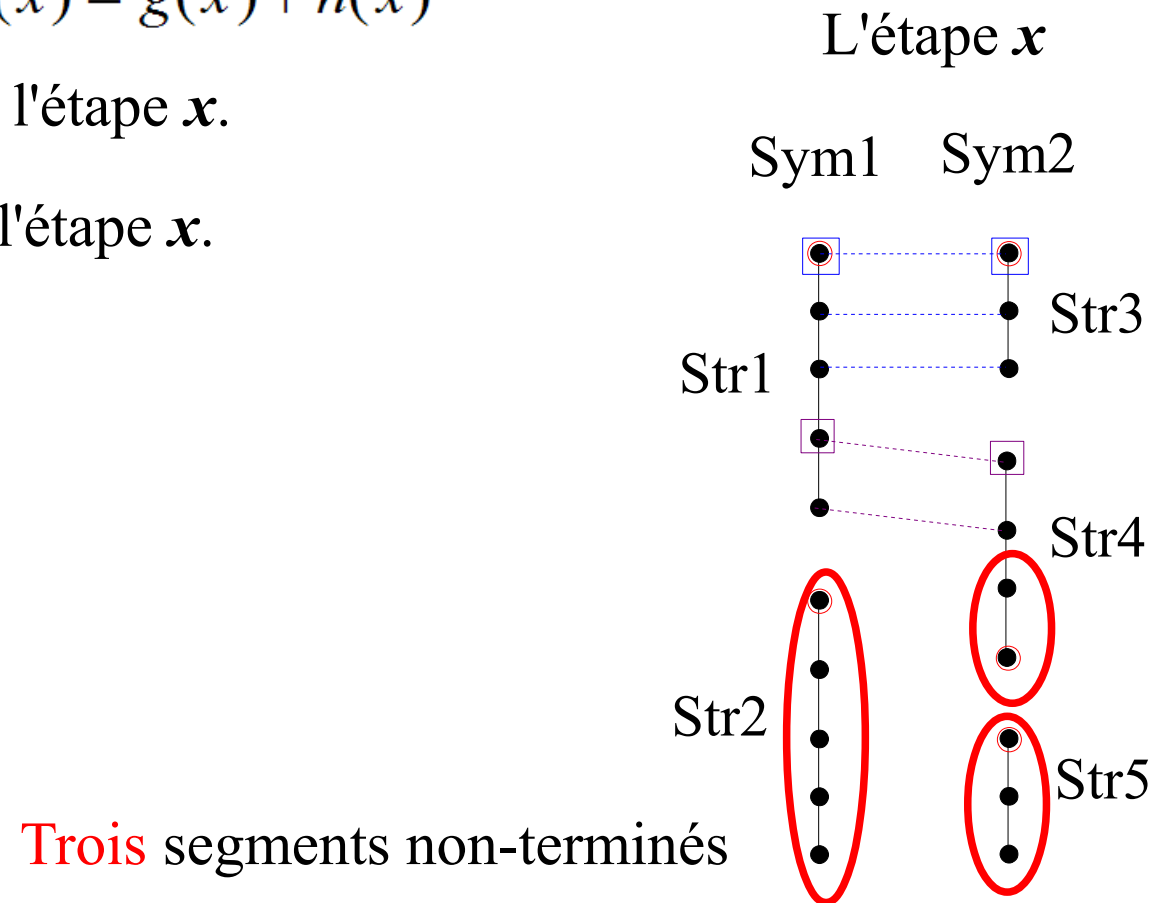
Le choix des points de départ

1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$

2: Le choix des points de départ à l'étape x .

- Les segments non-terminés à l'étape x .



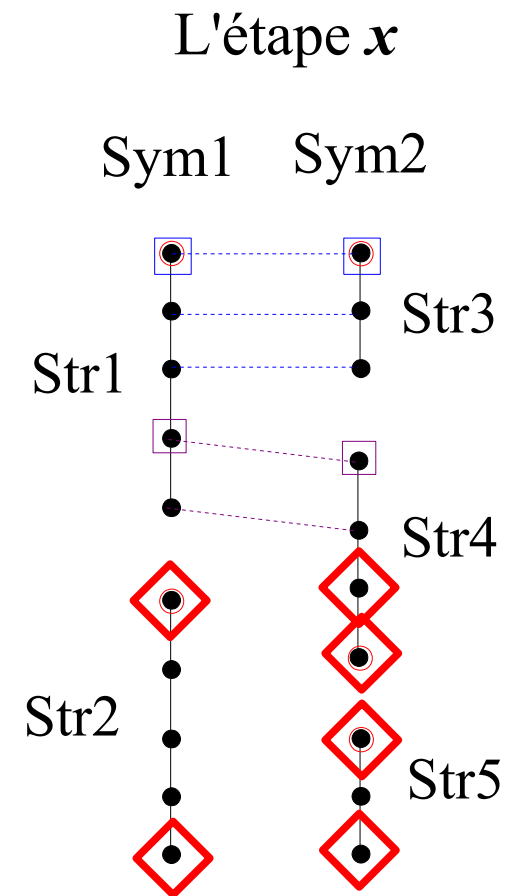
Le choix des points de départ

1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$

2: Le choix des points de départ à l'étape x .

- Les segments non-terminés à l'étape x .
- Les points de frontière de ces segments.



Le choix des points de départ

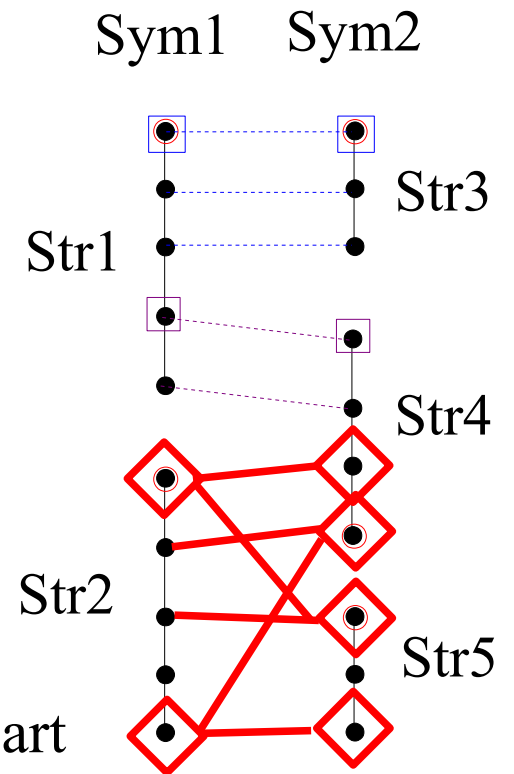
1: Pour accélérer la recherche, l'algorithme de recherche A* utilise une évaluation heuristique:

$$f(x) = g(x) + h(x)$$

2: Le choix des points de départ à l'étape x .

- Les segments non-terminés à l'étape x .
- Les points de frontière de ces segments.

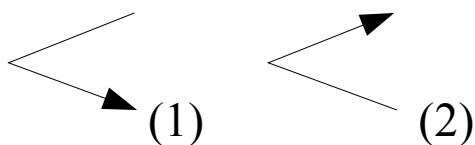
L'étape x



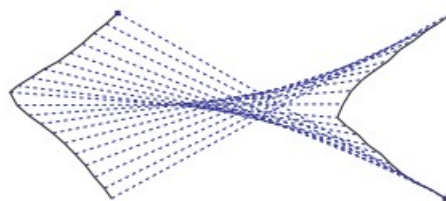
Étude expérimentale

Étude Qualitative

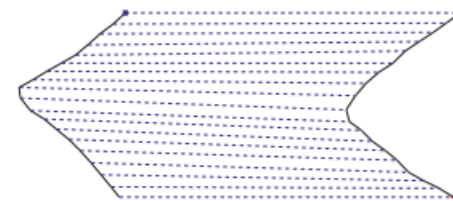
Les alignements entre deux exemples du symbole < dans différents sens, ordres et nombres de traits.



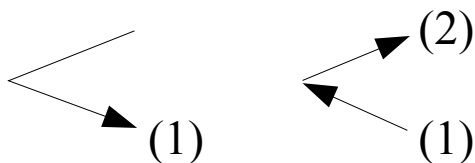
Cas 1



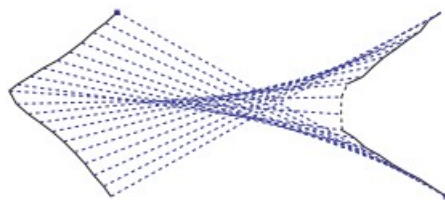
DTW Classique (distance=1.08)



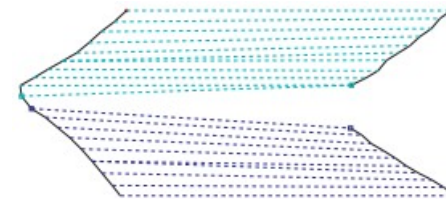
DTW A* (distance=0.1)



Cas 2



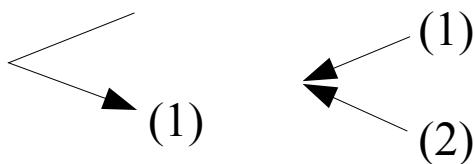
DTW Classique (distance=1.09)



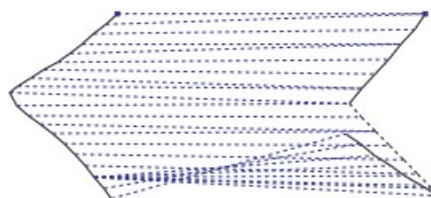
DTW A* (distance=0.11)

Étude Qualitative

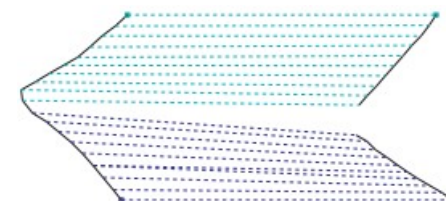
Les alignements entre deux exemples du symbole < dans différents sens, ordres et nombres de traits.



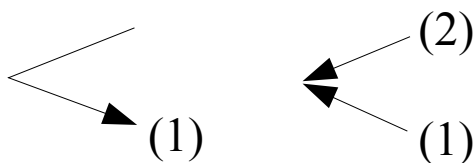
Cas 3



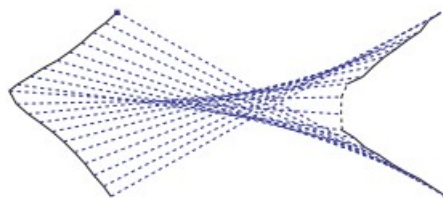
DTW Classique (distance=0.214)



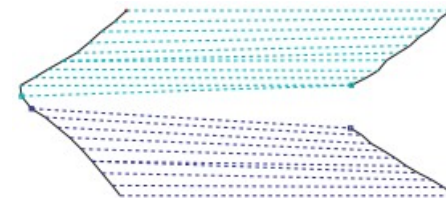
DTW A* (distance=0.15)



Cas 4



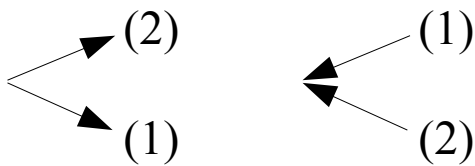
DTW Classique (distance=0.69)



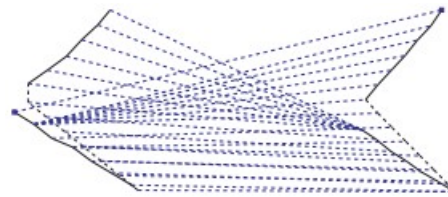
DTW A* (distance=0.10)

Étude Qualitative

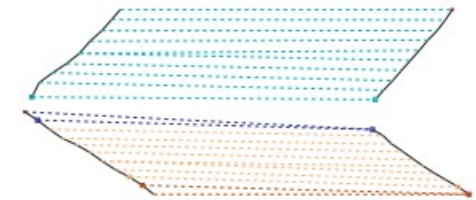
Les alignements entre deux exemples du symbole < dans différents sens, ordres et nombres de traits.



Cas 5



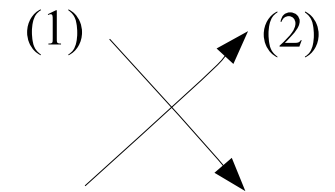
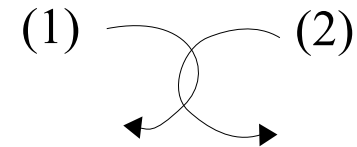
DTW Classique (distance=0.55)



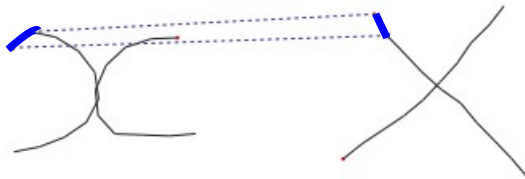
DTW A* (distance=0.1)

Étude Qualitative

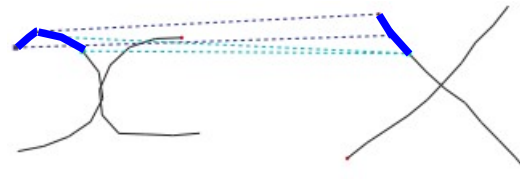
La meilleure solution entre deux \mathbf{x} :



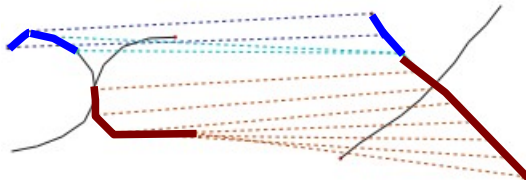
Etape 1



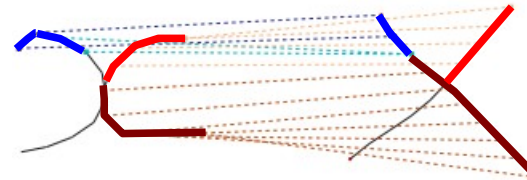
Etape 2



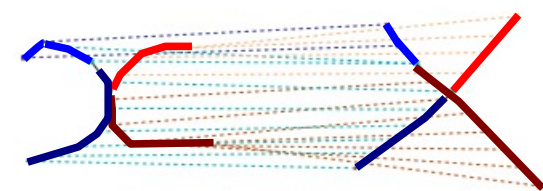
Etape 3



Etape 4

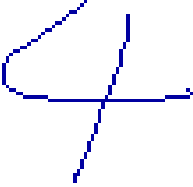
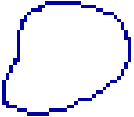
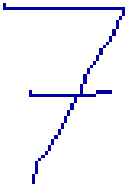
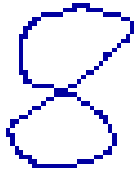

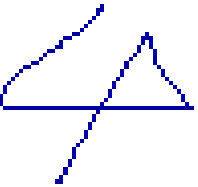


Etape 5



Étude Qualitative

Étude qualitative
pour évaluer la classification des caractères discriminants.

	 (2 str)	 (1 str)	 (2 str)	 (1 str)
 (2 str)	dist=0.29 temps=127 sec 77 304 hyp	dist=0.37 temps<1sec 3 749 hyp	dist=0.23 temps=787sec 238 193 hyp	dist=0.17 temps<1sec 218 hyp
 (1 str)	dist=0.15 temps<1 sec 112 hyp	dist=0.44 temps<1 sec 699 hyp	dist=0.27 temps=176 sec 88 820 hyp	dist=0.37 temps<1sec 16 hyp

Conclusion

- Définition d'une distance entre deux symboles multi-traits en respectant une contrainte de continuité
- Pour accélérer la recherche, l'algorithme de recherche A^* est utilisé.
- Première étude qualitative pour montre des intérêts.

Perspectives

- Faire un benchmark sur les données réelles.
- Détecter le grande nombre d'hypothèses

Merci de votre attention!
Questions?