

Nids de boucles

1 Analyse de dépendances

▷ **Question 1** Effectuer l'analyse de dépendances sur le code suivant (flot/anti/sortie) :

$$\begin{aligned} S_1 &: A \leftarrow B + C \\ S_2 &: D \leftarrow E + F \\ S_3 &: C \leftarrow A + C \\ S_4 &: E \leftarrow C + D \\ S_5 &: E \leftarrow 1 \end{aligned}$$

▷ **Question 2** Effectuer l'analyse de dépendance du programme suivant :

Pour $i = 1$ à N
Pour $j = 1$ à N
 $S_1 : a(i+1, j-1) \leftarrow b(i, j+4) + c(i-2, j-3) + 1$
 $S_2 : b(i-1, j) \leftarrow a(i, j) - 1$
 $S_3 : c(i, j) \leftarrow a(i, j-2) + b(i, j)$

2 Élimination de dépendances

▷ **Question 3** Calculer le GDR du code suivant :

Pour $i = 1$ à N
 $S_1 : a(i) \leftarrow b(i) + c(i)$
 $S_2 : a(i+1) \leftarrow a(i) + 2d(i)$

Essayer de casser le cycle de dépendance en introduisant une variable temporaire.

Dans le cas général, la découpe de sommet s'applique ainsi (notez que seules les utilisations avec dépendance de flot sont renommées en *temp*, sans quoi le code serait faux) :

Pour $i = 1$ à N
 \dots
 $S_k : lhs(f(i)) \leftarrow rhs(\dots)$
 \dots
 \dots
 $S_i : \dots \leftarrow lhs(g(i))$

avant la découpe

Pour $i = 1$ à N
 \dots
 $S'_k : temp(f(i)) \leftarrow rhs(\dots)$
 $S_k : lhs(f(i)) \leftarrow temp(f(i))$
 \dots
 $S_i : \dots \leftarrow temp(g(i))$

après la découpe

▷ **Question 4** On suppose que la fonction d'accès à lhs (pour left hand side) est injective. Montrer ce que deviennent les six types de dépendances possibles avec la découpe du sommet : dépendances en entrée sur S_k de type flot, anti et sortie, et dépendances en sortie de S_k de type flot, anti et sortie.

▷ **Question 5** Soit G le GDR d'un nid de boucles, et soit G' le graphe obtenu à partir de G en découplant tous ses sommets. Montrer qu'un cycle de G' comprend, soit uniquement des dépendances de flot, soit uniquement des dépendances de sortie. De plus, montrer que tout cycle de G' correspond à un cycle déjà présent dans G .

▷ **Question 6** Appliquer la méthode de découpe des sommets au code suivant (on découpera uniquement les sommets S_2 et S_3) :

Pour $i = 4$ à N

$$S_1 : a(i + 5) \leftarrow c(i - 3) + b(2i + 2)$$

$$S_2 : b(2i) \leftarrow a(i - 1) + 1$$

$$S_3 : a(i) \leftarrow c(i + 5) + 1$$

$$S_4 : c(i) \leftarrow b(2i - 4)$$

3 Algorithme d'Allen et Kennedy

▷ **Question 7** On considère le code suivant :

Pour $i = 1$ à N

Pour $j = 1$ à N

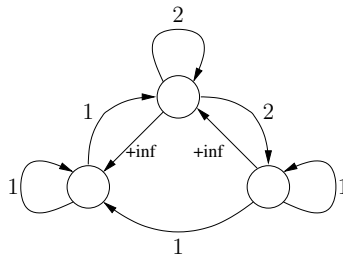
$$S_1 : a(i + 1, j + 1) \leftarrow a(i + 1, j) + b(i, j + 2)$$

$$S_2 : b(i + 1, j) \leftarrow a(i + 1, j - 1) + b(i, j - 1)$$

$$S_3 : a(i, j + 2) \leftarrow b(i + 1, j + 1) - 1$$

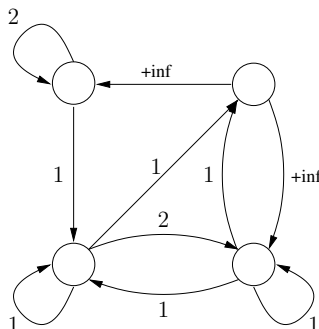
Donner le graphe de dépendances réduit, avec pour chaque dépendance son type (flot, anti, sortie) et son niveau. Appliquer l'algorithme d'Allen et Kennedy pour restructurer le nid de boucles et vérifier la nature des boucles obtenues.

▷ **Question 8** Donner un nid de boucles parfait où toutes les dépendances sont uniformes, et dont le GDRN est exactement le graphe suivant :



Exécuter l'algorithme d'Allen et Kennedy sur ce nid de boucles.

▷ **Question 9** Donner un nid de boucles parfait où toutes les dépendances sont uniformes, et dont le GDRN est exactement le graphe suivant :



Exécuter l'algorithme d'Allen et Kennedy sur ce nid de boucles.

4 Réponses aux exercices

▷ Question 1, page 1

Considérons la première instruction. Elle effectue une écriture sur A . La seule autre instruction qui utilise A est S_3 . Comme c'est une lecture et qu'elle intervient après S_1 , on a trouvé une dépendance de flot. En traitant ainsi toutes les instructions dans l'ordre, on obtient les dépendances suivantes :

- Flot $S_1 \rightarrow S_3$: variable A .
- Flot $S_2 \rightarrow S_4$: variable D .
- Anti $S_1 \rightarrow S_3$: variable C .
- Flot $S_3 \rightarrow S_4$: variable C .
- Anti $S_2 \rightarrow S_4$: variable E .
- Anti $S_2 \rightarrow S_5$: variable E (recouverte).
- Sortie $S_4 \rightarrow S_5$: variable E .

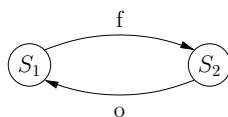
▷ Question 2, page 1

S_1 modifie la variable $a(i+1, j-1)$ et S_2 lit $a(i, j)$, il y a donc une dépendance entre ces deux instructions. Si on cherche une dépendance de $S_1(i, j)$ vers $S_2(i', j')$ (qui sera une dépendance de flot), on doit avoir $(i, j) <_{seq} (i', j')$ avec $i+1 = i'$ et $j-1 = j'$: on trouve bien une dépendance dont le vecteur de distance est $(1, -1)$. Si on avait cherché une dépendance de $S_2(i, j)$ vers $S_1(i', j')$ (nécessairement une ant dépendance), on aurait les équations $i = i' + 1$ et $j = j' - 1$ avec $(i, j) <_{seq} (i', j')$, c'est-à-dire $i < i'$ ou $i = i'$ et $j < j'$: pas de solution ici. Avec un peu d'habitude, on effectue la différence des indices et on oriente la dépendance dans la bonne direction, mais attention à ne pas se tromper ! Observons le cas de S_2 qui modifie $b(i-1, j)$ et de S_1 qui utilise $b(i, j+4)$. La première composante du vecteur d'itération devant être positive, on a une ant dépendance de S_1 vers S_2 . Au final, on obtient les dépendances suivantes :

- Flot $S_1 \rightarrow S_2$: variable a , distance $(1, -1)$
- Flot $S_1 \rightarrow S_3$: variable a , distance $(1, 1)$
- Flot $S_3 \rightarrow S_1$: variable c , distance $(2, 3)$
- Anti $S_1 \rightarrow S_2$: variable b , distance $(1, 4)$
- Anti $S_3 \rightarrow S_2$: variable b , distance $(1, 0)$

▷ Question 3, page 1

Le code initial comporte un dépendance de sortie de S_2 vers S_1 , car $a(i+1)$ est écrit dans S_2 à l'itération i avant d'être réécrit dans S_1 à l'itération suivante. Il y a également une dépendance de flot de S_1 vers S_2 à cause de $a(i)$. On obtient donc le GDR suivant :



L'ajout d'une variable temporaire permet de ■ découper ■ le sommet S_1 et d'éliminer le cycle :

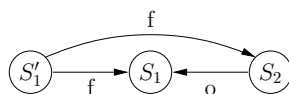
Pour $i = 1$ **to** N :

S'_1 : $temp(i) \leftarrow b(i) + c(i)$

S_1 : $a(i) \leftarrow temp(i)$

S_2 : $a(i+1) \leftarrow temp(i) + 2d(i)$

Le GDR du nouveau nid est le suivant :



Le cycle a été cassé. On peut distribuer la boucle pour obtenir le code parallélisé :

Pour $i = 1$ **to** N :

S'_1 : $temp(i) \leftarrow b(i) + c(i)$

$a(1) \leftarrow temp(1)$

Pour $i = 1$ **to** N :

S_2 : $a(i+1) \leftarrow temp(i) + 2d(i)$

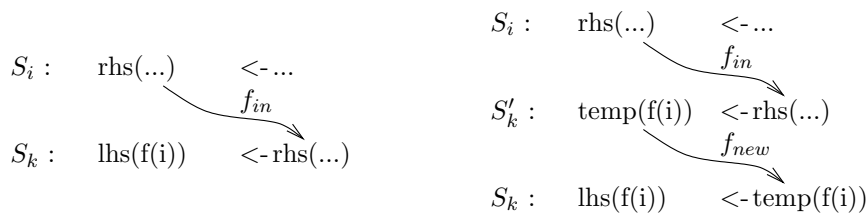


FIG. 1 – Dépendance de flot en entrée de S_k , avant et après la découpe de sommet.

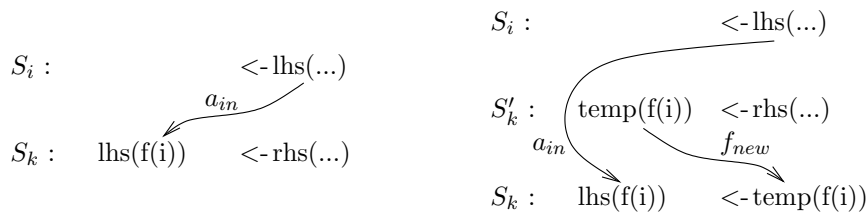


FIG. 2 – Antidépendance de flot en entrée de S_k , avant et après la découpe de sommet/

▷ **Question 4, page 1**

Il faut étudier successivement les six possibilités. On obtient les transformations suivantes :

- **Dépendance de flot en entrée de S_k** (figure 1). Supposons qu’une des données lues en membre droit de S_k a été produite par le membre gauche de S_i . Après la découpe du sommet, la donnée est lue en membre gauche de S'_k , la dépendance de flot est désormais dirigée vers S'_k . Il ne faut pas oublier la nouvelle dépendance de flot liée à $temp$ de S'_k vers S_k .
- **Antidépendance en entrée de S_k** (figure 2). Supposons qu’une instruction S_i lise $lhs(f(i))$ avant que S_k ne l’écrive. Après la découpe, $lhs(f(i))$ est toujours lue en S_k , l’antidépendance de S_i vers S_k est inchangée.
- **Dépendance de sortie en entrée de S_k** (figure 3). Supposons qu’une instruction S_i écrive $lhs(f(i))$ avant que S_k ne l’écrive. Après la découpe, cette dépendance de sortie de S_i vers S_k est inchangée.
- **Dépendance de flot en sortie de S_k** (figure 4). Supposons qu’une instruction S_i lise la valeur de $lhs(f(i))$ produite par S_k . Après la découpe, l’accès à $lhs(f(i))$ dans S_i a été remplacé par un accès à $temp(f(i))$, valeur produite par S'_k : il y a une dépendance de flot de S'_k vers S_i .
- **Antidépendance en sortie de S_k** (figure 5). Supposons qu’une des données lues en membre droit de S_k soit écrite après coup en membre gauche de S_i . Après la découpe, la donnée est lue en membre gauche de S'_k , l’antidépendance est désormais issue de S'_k vers S_i .
- **Dépendance de sortie en sortie de S_k** (figure 6). Supposons qu’une instruction S_i écrive $lhs(f(i))$ après que S_k l’a écrite. Après la découpe, cette dépendance de sortie de S_k vers S_i est inchangée.

Toutes ces transformations sont résumées figure 7.

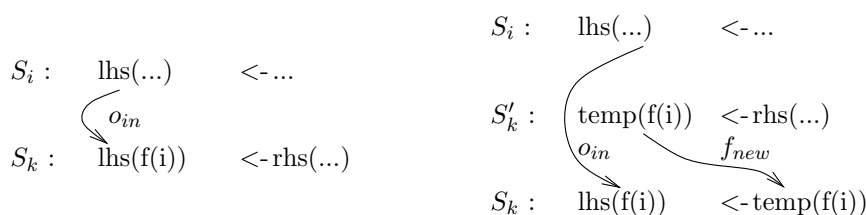


FIG. 3 – Dépendance de sortie en entrée de S_k , avant et après la découpe de sommet.

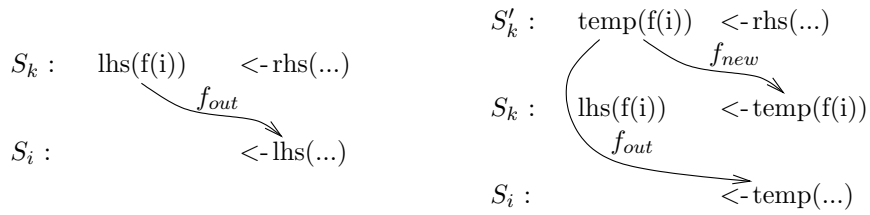


FIG. 4 – Dépendance de flot en sortie de S_k , avant et après la découpe de sommet.

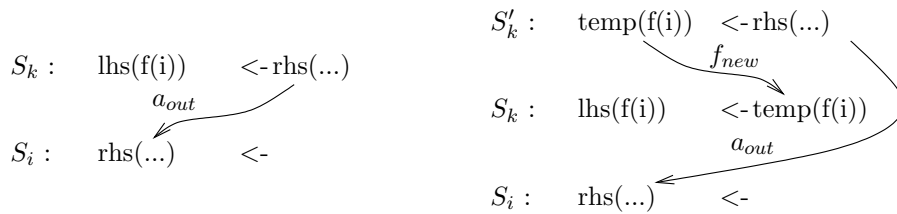


FIG. 5 – Antidépendance en sortie de S_k , avant et après la découpe de sommet.

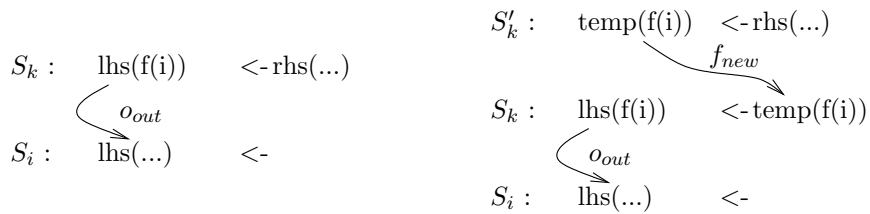


FIG. 6 – Dépendance de sortie en sortie de S_k , avant et après la découpe de sommet.

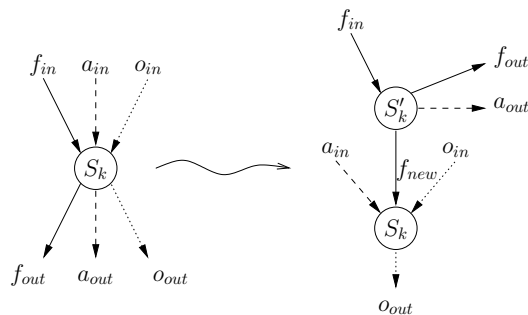


FIG. 7 – Transformations des dépendances lors de la découpe de sommet.

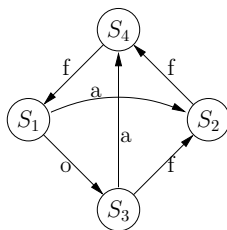


FIG. 8 – GDR avant la découpe de sommet.

▷ Question 5, page 1

La figure 7 permet de mieux suivre la preuve. Soit C un cycle de G' , et considérons une arête e de C :

- Si e correspond à une **dépendance de sortie**, alors d'après la figure 7, e est une arête d'un sommet S_k vers un sommet S_i : en entrée ou en sortie des nouveaux sommets S'_k il n'y a pas de dépendance de sortie. De même, les seules arêtes sortant de S_i correspondant à des dépendances de sortie, l'arête suivant e dans le cycle correspond, elle aussi, à une dépendance de sortie. Ainsi, C est uniquement composé d'arêtes représentant des dépendances de sortie. De plus, toutes les arêtes de C sont des arêtes qui étaient déjà présentes dans G .
- Si e correspond à une **antidépendance**, alors e va d'un sommet S'_k à un sommet S_i . Les seules arêtes sortant de S_i correspondant à des dépendances de sortie, l'arête suivant e dans le cycle correspond à une dépendance de sortie. En reprenant le raisonnement précédent, C est uniquement composé d'arêtes représentant des dépendances de sortie, ce qui est en contradiction avec la nature de e . Aucun cycle de G' ne peut contenir d'arête correspondant à une antidépendance.
- Si e correspond à une **dépendance de flot**, alors soit e a été créée lors de la découpe d'un sommet S_k , et relie S'_k à S_k , soit e correspond à une arête existante de S_k à S_i dans G et relie S'_k à S'_i :
 - $e : S'_k \xrightarrow{f_{new}} S_k$: les seules arêtes issues de S_k correspondant à des dépendances de sortie, l'arête suivant e dans C est une dépendance de sortie. Comme plus haut, C est composé uniquement de dépendances de sortie, contradiction.
 - $e : S'_k \xrightarrow{f} S'_i$: il peut y avoir des dépendances de flot ou des antidépendances issues de S'_i . Mais on a vu qu'il ne pouvait pas y avoir d'arête d'antidépendance dans un cycle, et l'arête qui suit e est nécessairement une arête représentant une dépendance de flot. Donc C est uniquement composé d'arêtes représentant des dépendances de flot. De plus, ces arêtes ne correspondent pas aux nouvelles dépendances introduites lors de la découpe des sommets, elles correspondent donc à des arêtes déjà présentes dans G .

En conclusion, la découpe des sommets n'a pas introduit de nouveau cycle de dépendance. Elle a permis de casser tous les cycles, sauf ceux composés uniquement de dépendances de flot et ceux composés uniquement de dépendances de sortie.

▷ Question 6, page 2

Le GDR est représenté figure 8.

Il y a six dépendances dans la boucle :

- trois dépendances de flot (de S_3 vers S_2 à cause de la variable a , de S_2 vers S_4 à cause de b et de S_4 vers S_1 à cause de c) ;
- deux antidépendances (de S_1 vers S_2 à cause de b , et de S_3 vers S_4 à cause de c) ;
- une dépendance de sortie (de S_1 vers S_3 à cause de a).

Le GDR est fortement connexe, et aucune distribution de boucle n'est possible. La découpe des sommets S_2 et S_3 conduit donc au GDR figure 9 :

Il n'y a plus de cycle dans le nouveau graphe. On peut réécrire le nid de boucles en introduisant les variables temporaires a_{temp} (pour la découpe de S_3) et b_{temp} (pour la découpe de S_2) :

Pour $i = 4$ to N :

$$S_1 : a(i + 5) \leftarrow c(i - 3) + b(2i + 2)$$

$$S'_2 : b_{temp}(2i) \leftarrow \mathbf{Si} \ i \geq 5 \ \mathbf{Alors} \ a_{temp}(i - 1) + 1 \ \mathbf{Sinon} \ a(i - 1) + 1$$

$$S_2 : b(2i) \leftarrow b_{temp}(2i)$$

$$S'_3 : a_{temp}(i) \leftarrow c(i + 5) + 1$$

$$S_3 : a(i) \leftarrow a_{temp}(i)$$

$$S_4 : c(i) \leftarrow \mathbf{Si} \ i \geq 6 \ \mathbf{Alors} \ b_{temp}(2i - 4) \ \mathbf{Sinon} \ b(2i - 4)$$

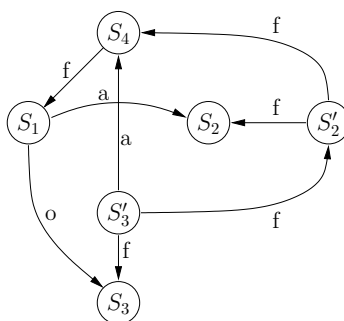


FIG. 9 – GDR après la découpe de sommet.

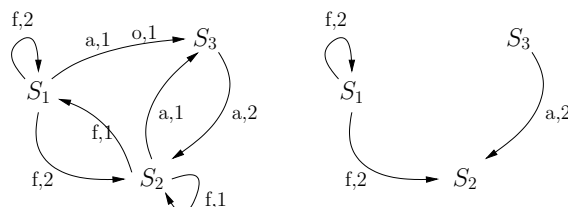
Les affectations conditionnelles sont nécessaires si on veut prendre en compte des dépendances provenant de l'extérieur du nid. Notons qu'on peut appliquer l'algorithme d'Allen et Kennedy sur le nouveau code pour distribuer puis paralléliser toutes les boucles (dans l'ordre topologique, par exemple $S'_3, S'_2, S_4, S_1, S_2, S_3$).

▷ **Question 7, page 2**

On obtient les dépendances suivantes :

- Flot $S_1 \rightarrow S_1$: variable a , distance $(0, 1)$
- Flot $S_1 \rightarrow S_2$: variable a , distance $(0, 2)$
- Flot $S_2 \rightarrow S_1$: variable b , distance $(1, -2)$
- Flot $S_2 \rightarrow S_2$: variable b , distance $(1, 1)$
- Anti $S_1 \rightarrow S_3$: variable a , distance $(1, -2)$
- Anti $S_2 \rightarrow S_3$: variable a , distance $(1, -3)$
- Anti $S_3 \rightarrow S_2$: variable b , distance $(0, 1)$
- Sortie $S_1 \rightarrow S_3$: variable a , distance $(1, -1)$

Le GDRN initial est dessiné sur la gauche :



Le GDRN est fortement connexe et il y a des dépendances de niveau 1. La boucle sur i sera donc séquentielle. En passant à la deuxième étape, on obtient le GDRN de droite sur la figure. L'algorithme d'Allen et Kennedy conduit alors au nid restructuré suivant :

```

Pour  $i = 1$  to  $N$  en séquentiel :
  Pour  $j = 1$  to  $N$  en séquentiel :
     $S_1 : a(i + 1, j + 1) \leftarrow a(i + 1, j) + b(i, j + 2)$ 
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_3 : a(i, j + 2) \leftarrow b(i + 1, j + 1) - 1$ 
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_2 : b(i + 1, j) \leftarrow a(i + 1, j - 1) + b(i, j - 1)$ 
    
```

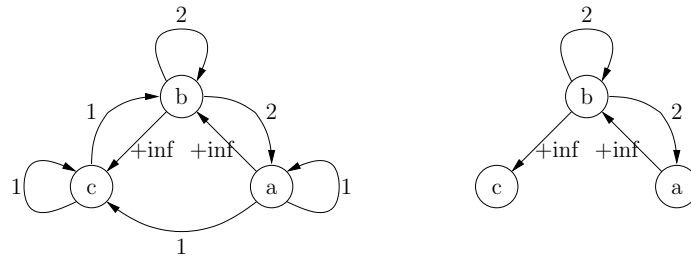
On notera qu'il est parfaitement possible d'intervertir la boucle portant S_3 avec celle portant S_1 .

▷ **Question 8, page 2**

Voici le nid de boucles :

```

Pour  $i = 1$  to  $N$  :
  Pour  $j = 1$  to  $N$  :
     $S_1 : a(i, j) \leftarrow a(i - 1, j) + b(i, j - 1)$ 
     $S_2 : b(i, j) \leftarrow a(i, j) + b(i, j - 1) + c(i - 1, j)$ 
     $S_3 : c(i, j) \leftarrow a(i - 1, j) + b(i, j) + c(i - 1, j)$ 
    
```



L'algorithme d'Allen et Kennedy conduit au nid restructuré suivant :

```

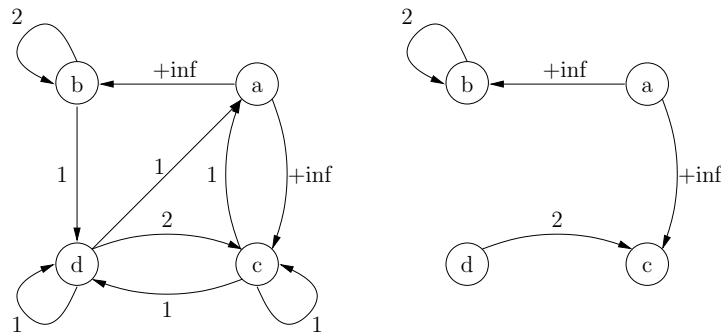
Pour  $i = 1$  to  $N$  en séquentiel :
  Pour  $j = 1$  to  $N$  en séquentiel :
     $S_1 : a(i, j) \leftarrow a(i - 1, j) + b(i, j - 1)$ 
     $S_2 : b(i, j) \leftarrow a(i, j) + b(i, j - 1) + c(i - 1, j)$ 
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_3 : c(i, j) \leftarrow a(i - 1, j) + b(i, j) + c(i - 1, j)$ 
    
```

▷ **Question 9, page 2**

On peut construire le code suivant :

```

Pour  $i = 1$  to  $N$  :
  Pour  $j = 1$  to  $N$  :
     $S_1 : a(i, j) \leftarrow d(i - 1, j) + c(i - 1, j)$ 
     $S_2 : b(i, j) \leftarrow a(i, j) + b(i, j - 1)$ 
     $S_3 : c(i, j) \leftarrow a(i, j) + c(i - 1, j) + d(i, j - 1)$ 
     $S_4 : d(i, j) \leftarrow c(i - 1, j) + d(i - 1, j) + b(i - 1, j)$ 
    
```



L'algorithme d'Allen et Kennedy conduit au nid restructuré suivant :

```

Pour  $i = 1$  to  $N$  en séquentiel :
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_4 : d(i, j) \leftarrow c(i - 1, j) + d(i - 1, j) + b(i - 1, j)$ 
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_1 : a(i, j) \leftarrow d(i - 1, j) + c(i - 1, j)$ 
  Pour  $j = 1$  to  $N$  en parallèle :
     $S_3 : c(i, j) \leftarrow a(i, j) + c(i - 1, j) + d(i, j - 1)$ 
  Pour  $j = 1$  to  $N$  en séquentiel :
     $S_2 : d(i, j) \leftarrow c(i - 1, j) + d(i - 1, j) + b(i - 1, j)$ 
    
```