ÉCOLE
POLYTECHNIQUE
MONTRÉAL

LOG3210
Cours 12

Allocation de registres

Allocation de registres simple

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Pour chaque instruction de type x = y + z, notre générateur de code doit stocker les opérandes (y et z) et le résultat (x) dans des registres. Pour ce faire il fera appel à la fonction *getReg*

- Lors d'un appel à *getReg* pour une opérande y, différentes situations peuvent survenir:
  1. L'opérande y est déjà dans un registre R_y. Retourne R_y.
  2. L'opérande y n'est pas dans un registre, mais il y a un registre R_y libre. Retourne R_y (le générateur de code se charge de générer un (LD R_y y) si nécessaire).
  3. L'opérande y n'est pas dans un registre et aucun registre n'est libre. Il faut choisir un registre à libérer…

Allocation de registres simple (suite)

ÉCOLE POLYTECHNIQUE MONTRÉAL

3. L'opérande y n'est pas dans un registre et aucun registre n'est libre. Supposons un registre R que nous voudrions libérer et *v* une des variables contenue dans R. Les possibilités sont:
   a. Si *v* est aussi contenue dans un autre registre ou une autre adresse mémoire, alors OK.
   b. Si *v* reçoit le résultat de l'instruction (v est x) et *v* n'est pas une opérande (v n'est ni y ni z) alors OK.
   c. Si *v* n'est pas utilisée ultérieurement après l'instruction que nous traitons (information *next-use*), alors OK.
   d. Si aucune des étapes précédentes ne retourne OK, il faut stocker v dans son adresse mémoire (ST *v*, R). Cette opération s'appelle un déversement (*spill*).

Allocation de registres simple (suite)

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Puisque R peut contenir plus d'une variable, il faut répéter les étapes précédentes pour chaque variable *v* contenue dans R.

- Le nombre d'instructions de stockage (voir étape 3.d) à générer constitue le « score » d'un registre. On libère le registre avec le score le plus bas.

Allocation de registres simple (suite)

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Lors d'un appel à *getReg* pour le résultat x, le traitement est très similaire à celui pour une opérande y, à quelques exceptions près:
  - À l'étape 3.b, si *v* est x, alors OK même si x est aussi une opérande.
  - À l'étape 3.c, si R est R_y, et que y n'est plus utilisé par la suite, il est possible d'utiliser R à la fois pour R_y et R_x.

- Finalement, pour des instructions de copie (x = y), on choisit R_y de la même manière, mais R_x est toujours égal à R_y.

Allocation de registres simple (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

	R1	R2	R3	a	b	c	d	e

(1) a = b + c
LD R1, b
LD R2, c
ADD R1, R2, R3
	b	c	a	R3	b
					R1
						c
						R2
							d	e

Il faut choisir un registre pour *d*. - R1 : contient *b*, next-use (2). - R2 : contient *c*, next-use (3). - R3 : contient *a*, next-use (3), *spill* requis.								

(2) d = d - b
LD R2, d
SUB R2, R1, R2
	b	d	a	R3	b
					R1
						c	R2	e

(3) e = a + c
LD R1, c
ADD R3, R1, R1
	~~e~~
	e
		d	a	R3	b	c	R2	R1

Exit: Stocker les variables **vives** contenues dans un registre (*ST*).								

Allocation de registres par coloriage de graphe

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Nous avons vu qu'il est parfois nécessaire de déverser (spill) le contenu d'un registre en mémoire.
- L'algorithme que nous avons vu, bien que fonctionnel, n'est pas très efficace pour minimiser le nombre de déversements.
- Une heuristique par coloriage de graphe donne généralement de meilleurs résultats.

Allocation de registres par coloriage de graphe

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Nous débuterons en supposant que nous avons une quantité infinie de registres.

- Au moment de transformer le code à 3 adresses en code machine, nous supposerons que chaque variable possède son propre registre. Ce seront des registres *symboliques*.

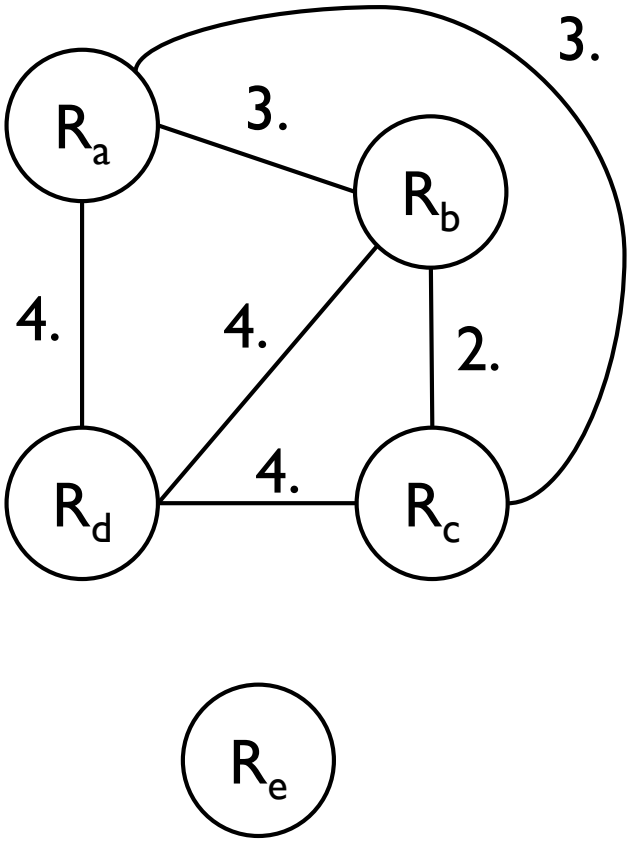
- Nous construirons ensuite un graphe d'interférence de registres où les nœuds représenteront des registres symboliques (variables) et où les arêtes relieront deux variables si l'une est vive lorsque l'autre est définie.

Allocation de registres par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

```
a = b + c
d = d - b
e = a + c
```

```
1. LD     Rb, b
2. LD     Rc, c
3. ADD    Ra, Rb, Rc
4. LD     Rd, d
5. SUB    Rd, Rd, Rb
6. ADD    Re, Ra, Rc
```

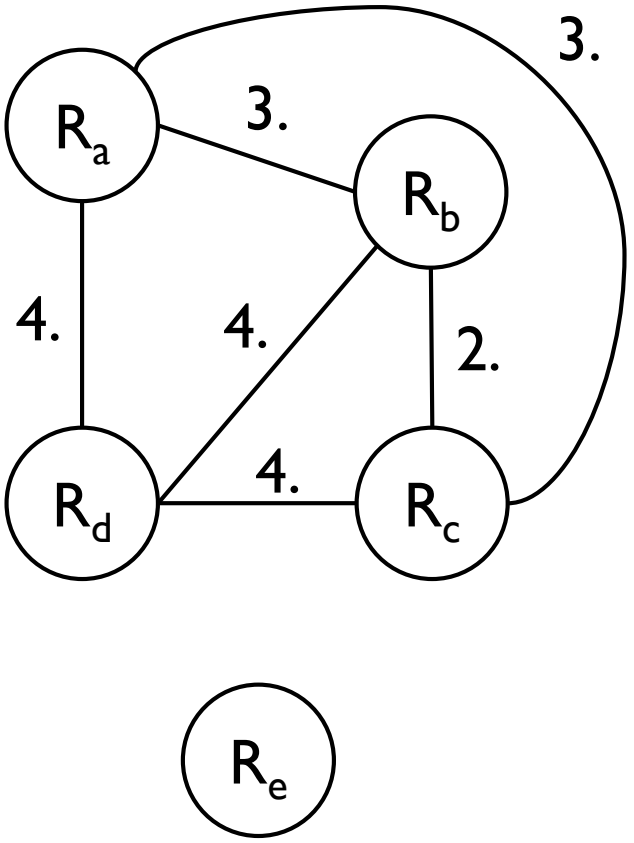


Allocation de registres par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

```
a = b + c
d = d - b
e = a + c
```

Instructions	*Liveness*	*Next use*
1. LD Rb, b	-	
2. LD Rc, c	b	b3
3. ADD Ra, Rb, Rc	b, c	b3, c3
4. LD Rd, d	a, c, b	a6, c6, b5
5. SUB Rd, Rd, Rb	a, c, d, b	a6, c6, d5, b5
6. ADD Re, Ra, Rc	a, c	a6, c6
End	-	



Allocation de registres par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

- Le but est de colorier le graphe avec *k* couleurs, où *k* est le nombre de registres, de telle sorte qu'il n'y ait pas de voisins qui aient la même couleur.

- Si nous y arrivons, nous avons trouvé une manière correcte d'allouer les registres. Le *k*-coloriage est NP-complet.

Heuristique d'allocation par coloriage de graphe (suite)

ÉCOLE POLYTECHNIQUE MONTRÉAL

1. **Simplifie:** Retirer et empiler un nœud *n* qui a moins de *k* voisins. Répéter tant qu'il est possible de retirer un nœud.
   Note: Si le graphe résultant G' = G – {*n*} peut être *k*-colorié, alors G peut être *k*-colorié puisque *n* a moins de *k* voisins.

2. **Déverse:** S'il n'est pas possible de retirer un nœud, un déversement (*spill*) est nécessaire. On prend un nœud restant, on le marque comme déversement potentiel, on le retire et on l'empile.

3. **Sélectionne:** Quand le graphe est vide, on retire un à un les nœuds de la pile et on les réintroduit dans le graphe en leur attribuant une couleur. Si le nœud réintroduit est un déversement potentiel, on essaie quand même de le colorier. Si c'est impossible, nous avons un déversement réel. Dans ce cas, le nœud n'est pas colorié et l'algorithme continue.

Heuristique d'allocation par coloriage de graphe

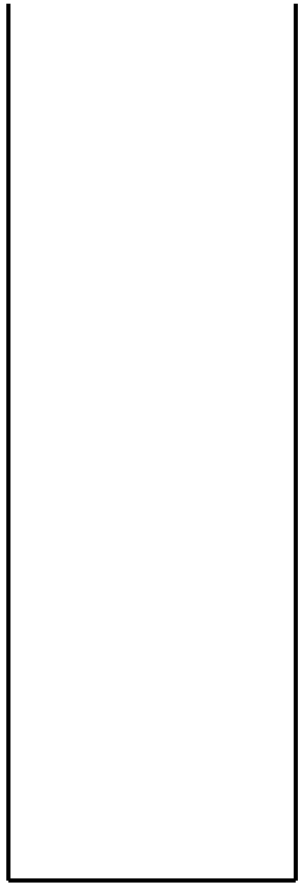
ÉCOLE POLYTECHNIQUE MONTRÉAL

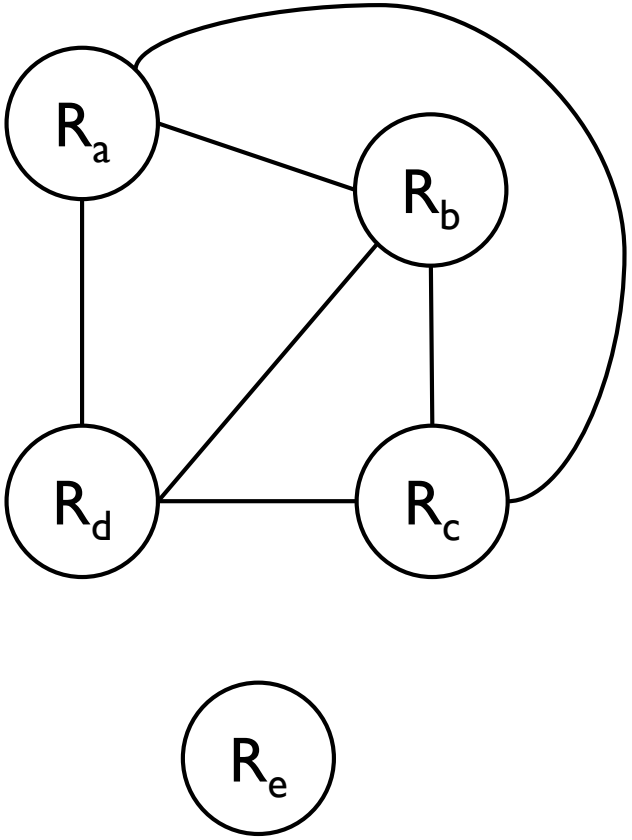
4. **Recommence:** S'il y a au moins un nœud qui n'a pu être colorié, le code doit être ajusté pour ajouter des instructions qui chargeront le nœud dans un nouveau temporaire avant chaque usage (LD) et qui stockeront le temporaire après chaque définition (ST).
   Ces modifications impactent le graphe d'interférence de registres qui doit être mis à jour avant de relancer l'étape **simplifie**.

Heuristique d'allocation par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

Supposons k = 3

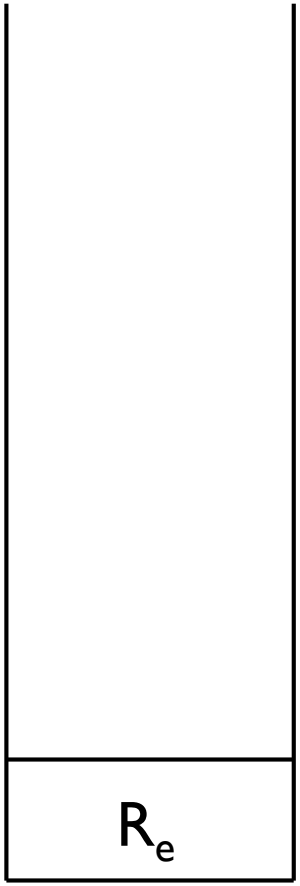


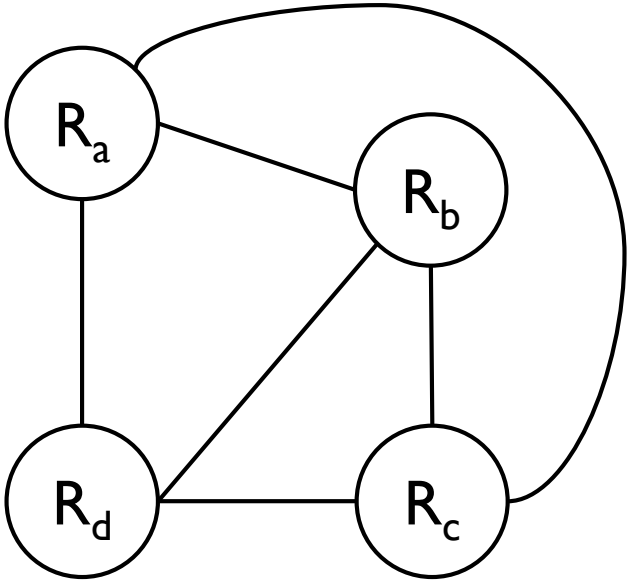


Heuristique d'allocation par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

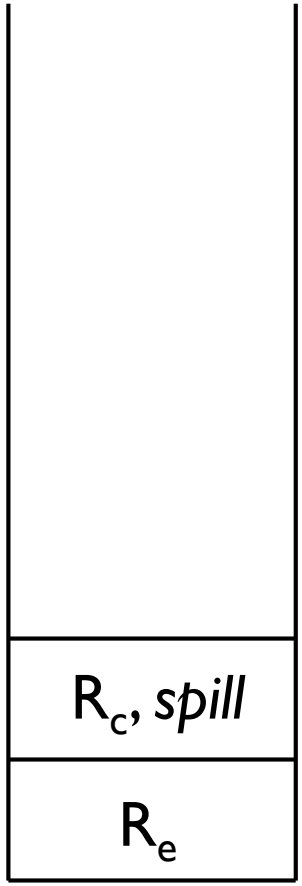
Supposons k = 3

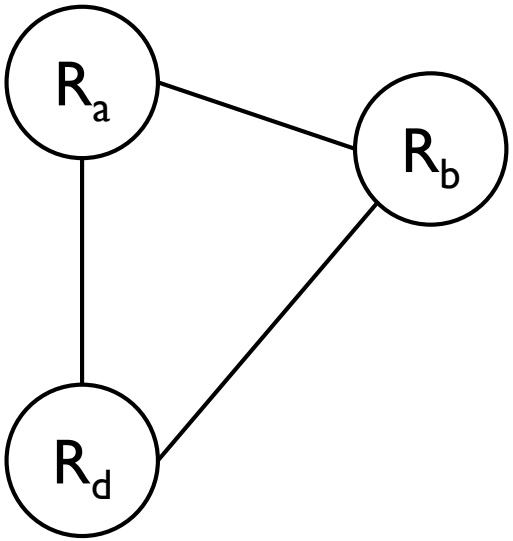




Heuristique d'allocation par coloriage de graphe (exemple)

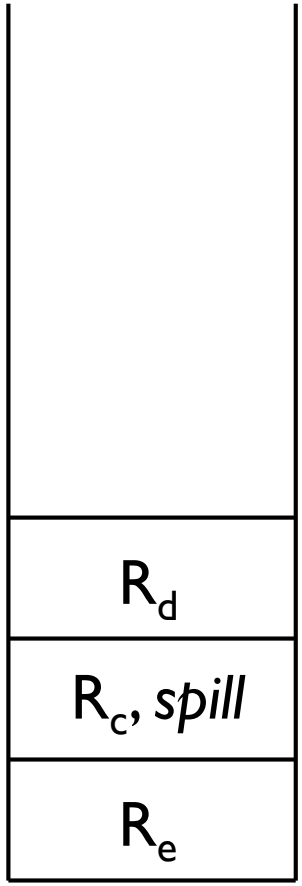
Supposons k = 3

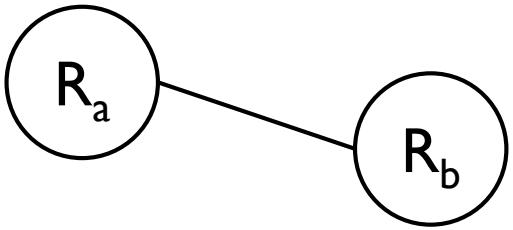




Heuristique d'allocation par coloriage de graphe (exemple)

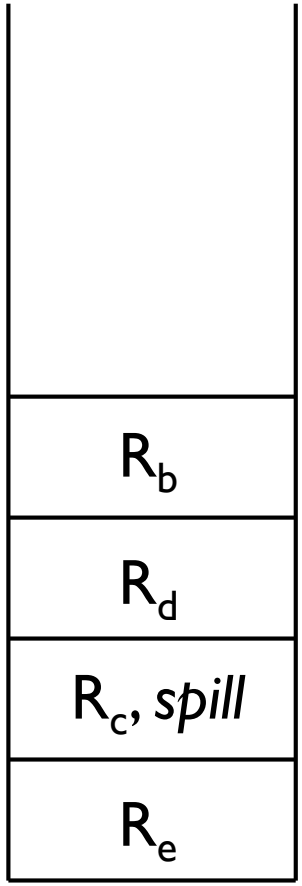
Supposons k = 3

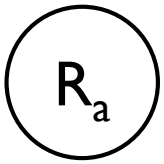




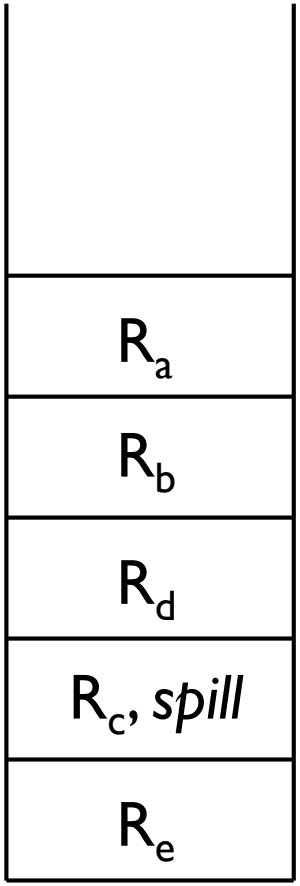
Heuristique d'allocation par coloriage de graphe (exemple)

Supposons k = 3





Heuristique d'allocation par coloriage de graphe (exemple)

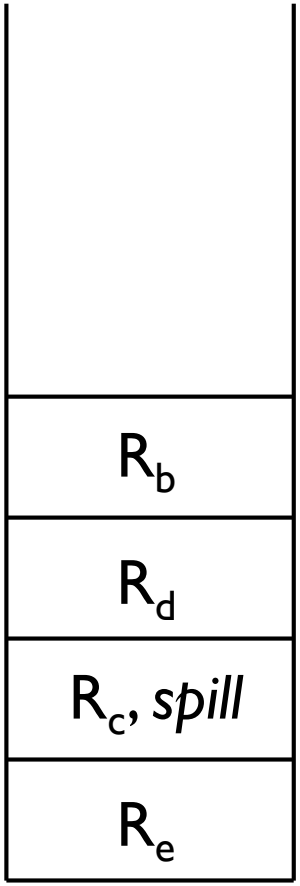


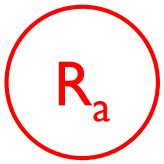
Supposons k = 3

Simplifie est terminée

Heuristique d'allocation par coloriage de graphe (exemple)

Supposons k = 3

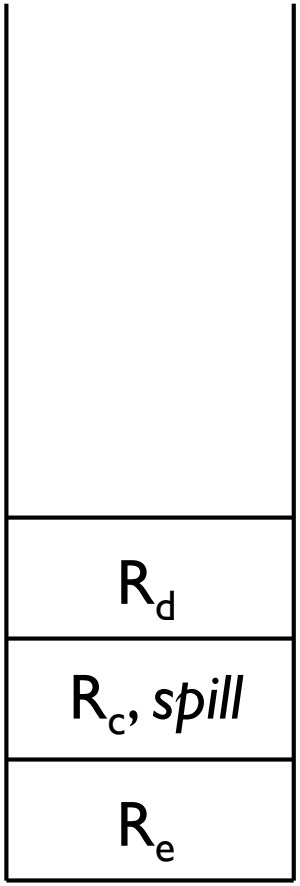


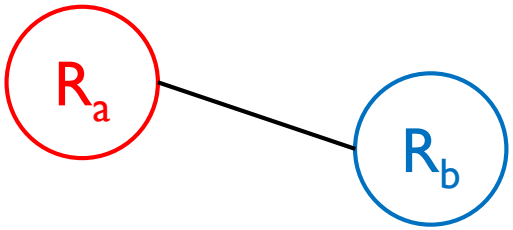


Heuristique d'allocation par coloriage de graphe (exemple)

ÉCOLE POLYTECHNIQUE MONTRÉAL

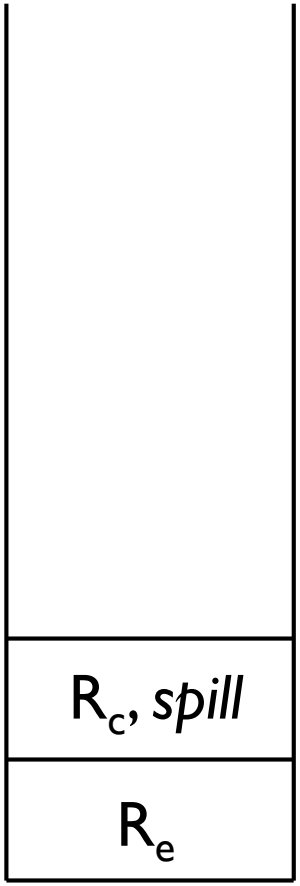
Supposons k = 3

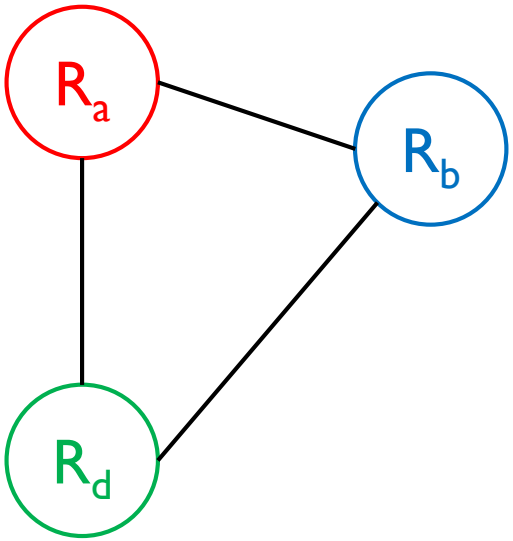




Heuristique d'allocation par coloriage de graphe (exemple)

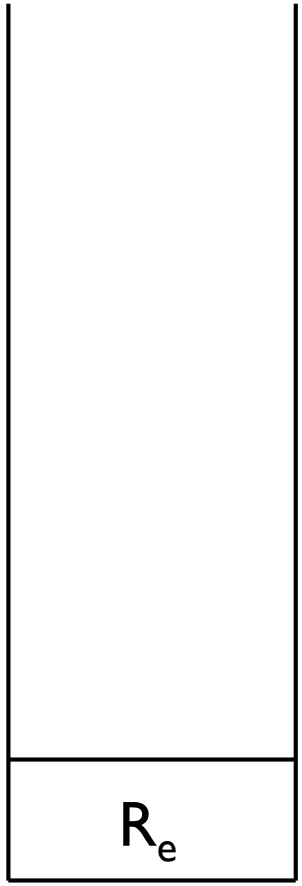
Supposons k = 3

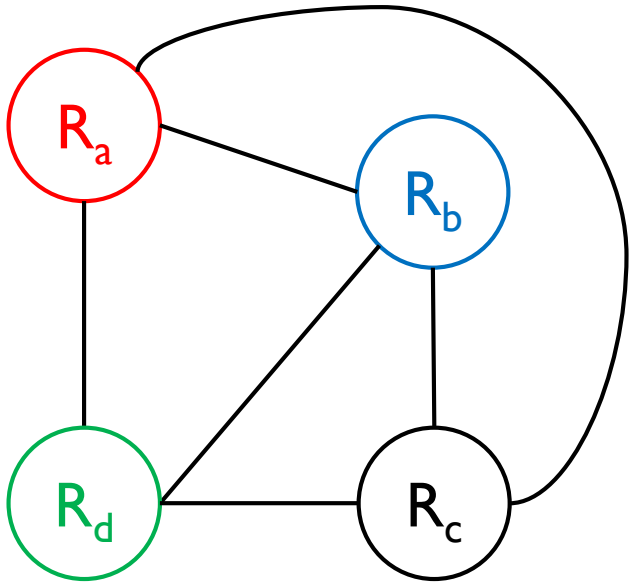




Heuristique d'allocation par coloriage de graphe (exemple)

Supposons k = 3

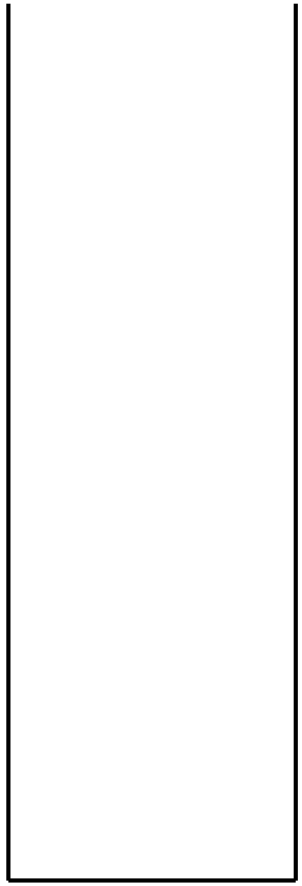




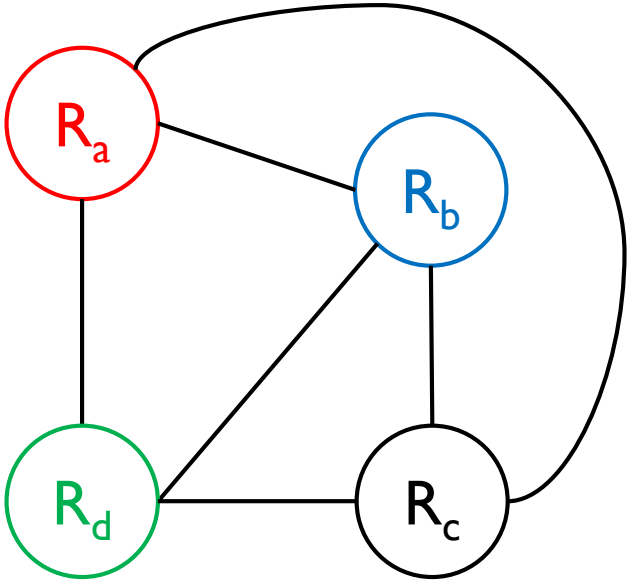
spill réel

Heuristique d'allocation par coloriage de graphe (exemple)

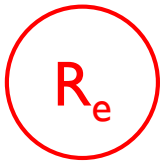
Supposons k = 3



Sélectionne est terminée

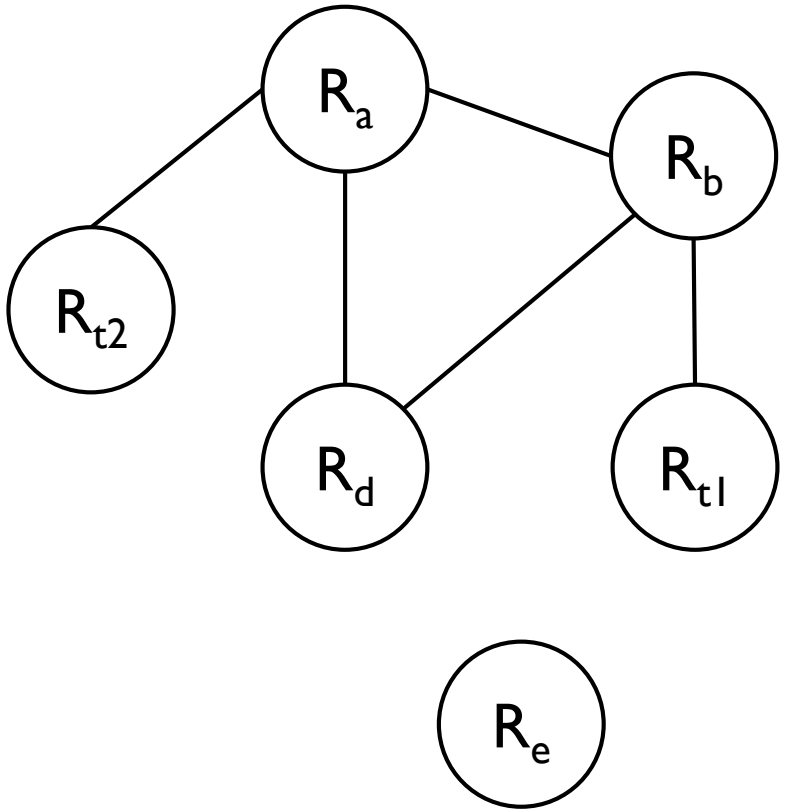


spill réel



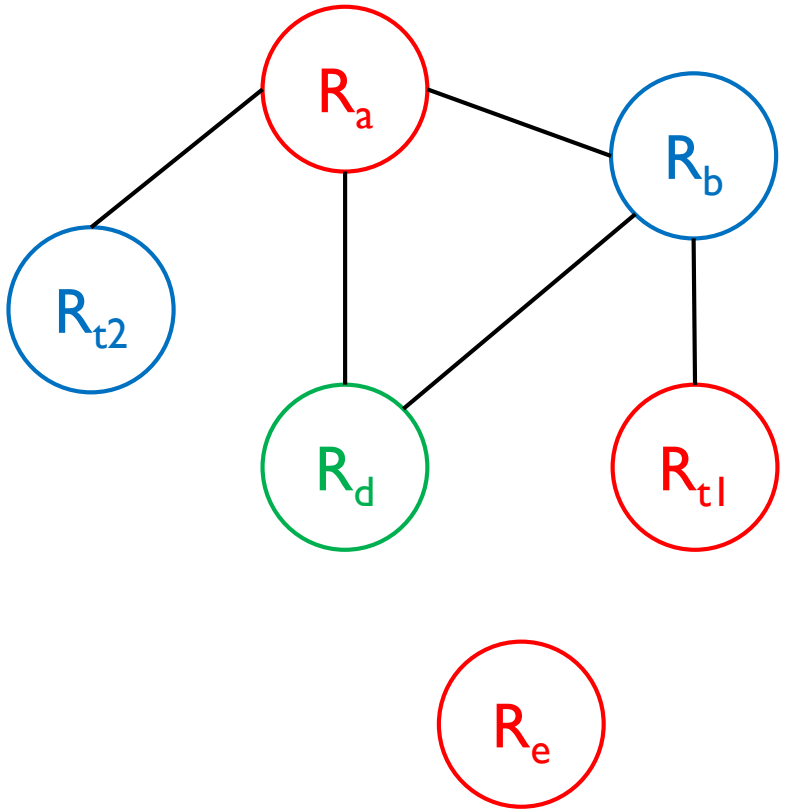
Code ajusté pour le déversement

1. LD R_b, b
2. ~~LD R_e, c~~
2. **LD R_{t1}, c**
3. ADD R_a, R_b, **R_{t1}**
4. LD R_d, d
5. SUB R_d, R_d, R_b
6. **LD R_{t2}, c**
7. ADD R_e, R_a, **R_{t2}**



Coloriage possible

1. LD R_b, b
2. ~~LD R_e, c~~
2. **LD R_{t1}, c**
3. ADD R_a, R_b, **R_{t1}**
4. LD R_d, d
5. SUB R_d, R_d, R_b
6. **LD R_{t2}, c**
7. ADD R_e, R_a, **R_{t2}**



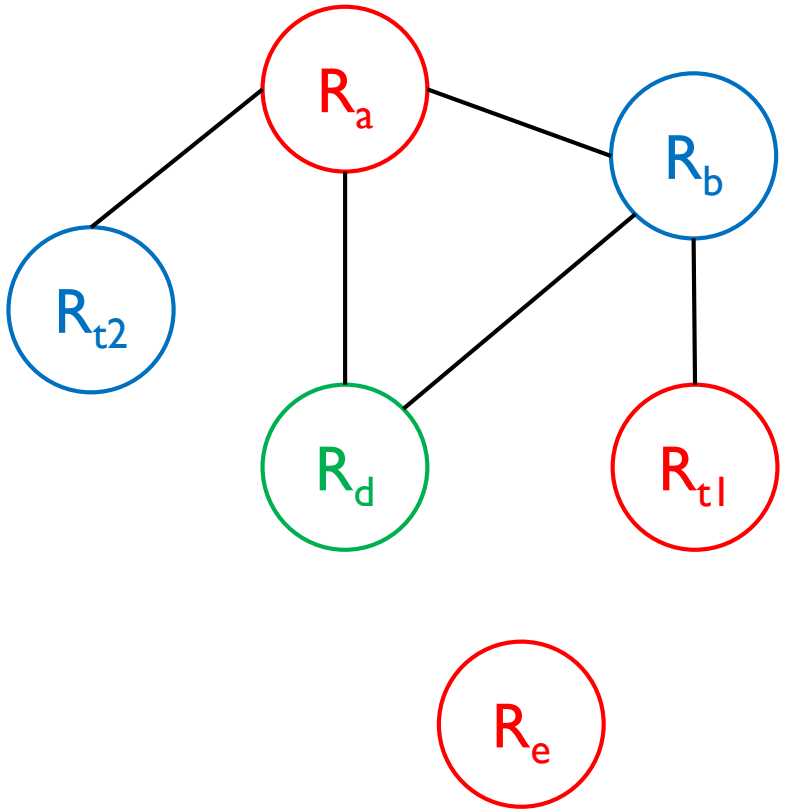
Code final

```
1. LD      R1, b
2. LD      R2, c
3. ADD     R2, R1, R2
4. LD      R3, d
5. SUB     R3, R3, R2
6. LD      R1, c
7. ADD     R2, R2, R1
```

Bleu = R_1

Rouge = R_2

Vert = R_3



Allocation de registres par coloriage de graphe

ÉCOLE POLYTECHNIQUE MONTRÉAL

- L'algorithme de coloriage que nous avons présenté est relativement simple.
- Plusieurs heuristiques existent pour sélectionner les nœuds qui seront déversés.
- Par exemple, en présence de boucles, on essaiera généralement de ne pas déverser les nœuds qui se trouvent dans les boucles imbriquées.
- Voir le livre de Appel (livre du Tigre) pour du matériel plus avancé.