

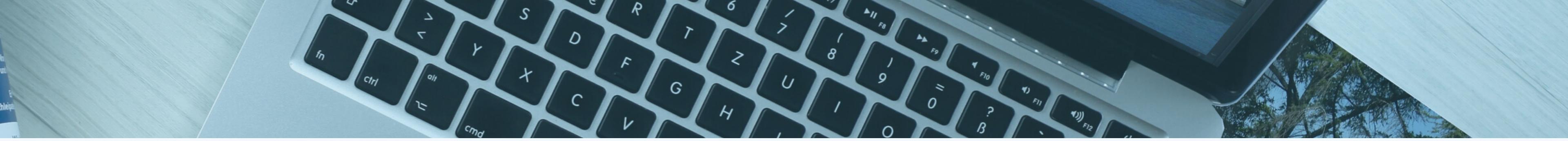
SYSTÈME D'EXPLOITATION

PARALLÉLISATION MAXIMALE AUTOMATIQUE

Presenté par Seycha Oudeinguene
Mupuanga-Lumbanzila Jeancy
Pisano Vincent

SOMMAIRE

Histoire du projet
Présentation
Exécution du code
Conclusion



HISTOIRE DU PROJET

DÉVELOPPER UNE LIBRAIRIE EN PYTHON

Objectifs :

- 1 - Créer un système de tâches de parallélisme maximal à partir des tâches en entrée. ✓
- 2 - Exécuter le système de tâches en parallèle, en respectant les contraintes du parallélisme maximal. ✗
- 3 - Vérifier la validation des entrées fournies à la procédure de construction du système de tâches. ✓
- 4 - Créer et afficher graphique le graphe de précédence du système de parallélisme maximal. ✓

PRÉSENTATION

STRUCTURE PROJET

Maxpar.py
Class Task
Class MonThread
Class TaskSystem

Class Task

NAME

Un String répertoriant
le nom de la tâche

READS

Un tableau de String
répertoriant les tâches
que lit cette tâche

WRITES

Un tableau de String
répertoriant les tâches
où cette tâche écrit

RUN

Une variable qui stocke
le nom de la fonction
qu'exécute cette tâche

Class MonThread

INIT

C'est une méthode appelée quand un objet est créé à partir d'une classe et elle permet à la classe d'initialiser les attributs de la classe.

RUN

Exécute les tâches du système en parallélisant celles qui peuvent être parallélisées selon la spécification du parallélisme maximal.

Class TaskSystem

INIT

Méthode qui initialise la liste des tâches, les préférences ainsi que les préférences Finales.
Créer et affiche le graphe

GETDEPENDENCIES

Méthode Alimentant les préférences Finales ainsi que le graphe avec les préférences vérifiant les conditions de Bernstein.

RUN

Méthode qui permet d'appeler la classe MonThread sur les tâches qui peuvent être parallélisées.

INTERFERENTE

Méthode implémentant les conditions de Bernstein et qui retourne True s'il y a une interférence et False si non.

EXÉCUTION DU CODE

The screenshot shows the Visual Studio Code interface with the title bar "maxpar.py - Untitled (Workspace) - Visual Studio Code". The left sidebar includes icons for Explorer, Search, Open, and Recent. The Explorer view shows a workspace with files: "Projet", "demo_dot.png", "demo.dot", "demo.dot.pdf", "maxpar.pdf", and "maxpar.py" (which is currently selected). The main editor area displays Python code for a task-based parallel execution system:

```
Projet > maxpar.py > ...
112     global X, Y, Z
113     Z = X + Y
114
115
116     t1 = Task()
117     t1.name= "T1"
118     t1.writes= ["X"]
119     t1.run= runT1
120
121     t2 = Task()
122     t2.name= "T2"
123     t2.writes= ["X"]
124     t2.run= runT2
125
126     tSomme= Task()
127     tSomme.name= "somme"
128     tSomme.reads= ["X", "Y"]
129     tSomme.writes= ["Z"]
130     tSomme.run = runTSomme
131
132     t3 = Task()
133     t3.name= "T3"
134     t3.writes= ["Z"]
135     t3.reads= ["Y"]
136     t3.run= runT2
137
138
139     t1.run()
140     t2.run()
141     tSomme.run()
142     #print(X)
143     #print(Y)
144     #print(Z)
145
```

The bottom navigation bar includes tabs for "OUTPUT", "TERMINAL", "DEBUG CONSOLE", and "PROBLEMS". A context menu is open over the code, listing various actions:

- Run Code (Ctrl+Alt+N)
- Go to Definition (F12)
- Go to References (Shift+F12)
- Peek (with a dropdown arrow)
- Find All References (Shift+Alt+F12)
- Rename Symbol (F2)
- Change All Occurrences (Ctrl+F2)
- Format Document (Shift+Alt+F)
- Format Document With... (dropdown)
- Source Action... (dropdown)
- Cut (Ctrl+X)
- Copy (Ctrl+C)
- Paste (Ctrl+V)
- Run Current File in Interactive Window
- Run From Line in Interactive Window
- Run Selection/Line in Interactive Window (Shift+Enter)
- Run To Line in Interactive Window
- Run Current Test File
- Run Python File in Terminal
- Run Selection/Line in Python Terminal (Shift+Enter)

SYSTÈME D'EXPLOITATION

CONCLUSION

Présenté par Seycha Oudeinguene
Mupuanga-Lumbanzila Jeancy
Pisano Vincent

QUESTIONS?