**Informe Final Projecte**

**Programació de tasques eficients en centres de dades per a estalvi energètic**

**Índex**

[Introducció i Contextualització 3](#_7cu1xxge78f)

[Metodologia 4](#_892d72h9s61c)

[Modelatge Matemàtic 4](#_uhowdbfo8cv2)

[Funció Objectiu: 4](#_4rthoqpb2pf3)

[Restriccions: 4](#_56b9pcx6t5ze)

[Generació de dades 5](#_5jk6yqjpymjt)

[Implementació 6](#_rvm5f1b5gygn)

[Resultats i Discrepàncies 7](#_5tv5jmpwh93l)

[Resultats de la Implementació 7](#_wa8gmziynbdv)

[Propostes de millora 8](#_2fb3nvt443u8)

[Conclusió i Treball Futur 9](#_ynz7w3g6tqv8)

## 

## **Introducció i Contextualització**

L’optimització del consum energètic és un repte clau en sistemes computacionals moderns, especialment en l'àmbit de la computació d’alt rendiment, on l’assignació eficient de tasques als processadors pot reduir significativament els costos energètics. Aquest projecte aborda aquest repte mitjançant la modelització matemàtica i la implementació d’un model d’optimització en Python.

L’objectiu és minimitzar el consum energètic, complint amb les restriccions de temps d'execució i capacitat dels processadors. Per assolir-ho, es defineix una funció objectiu i es modelen les restriccions corresponents.

El projecte es pot visualitzar en el repositori de GitHub: <https://github.com/luciarevaliente/Efficient-Task-Scheduling-in-Data-Centers-for-Energy-Savings.git>.

## 

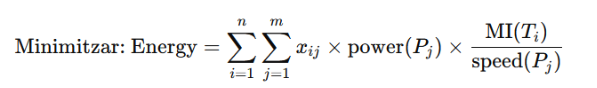
## **Metodologia**

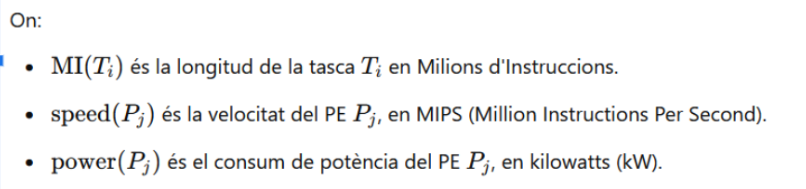
### **Modelatge Matemàtic**

A continuació es mostra la formulació del Modelatge Matemàtic utilitzat per resoldre el problema d'assignació de tasques a processadors, amb l'objectiu de minimitzar el consum energètic. Aquí tens una explicació més clara i detallada de cada element.

#### **Funció Objectiu:**

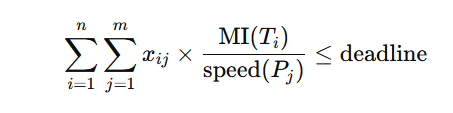
L'objectiu és minimitzar el consum energètic total en assignar tasques als processadors. La funció objectiu es defineix com:



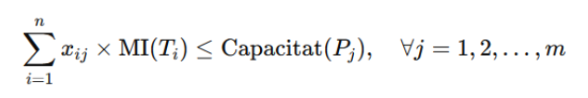


#### **Restriccions:**

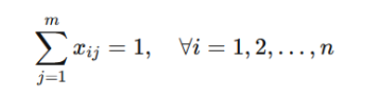
1. **Limitació del temps d’execució:** Cada tasca ha de completar-se dins del seu **deadline**. Això es tradueix en la següent inequació.



1. **Capacitat del processador:** La càrrega total assignada a un processador no pot superar la seva capacitat màxima.



1. **Assignació única de tasques:** on Xij = [0,1]. Cada tasca ha de ser assignada exactament a un processador. Aquesta restricció s'expressa com:



### **Generació de dades**

En el desenvolupament d'aquest projecte, hem hagut de generar les dades necessàries ja que no existia cap dataset oficial relacionat amb el problema plantejat. El codi per generar les dades es troba a l'arxiu create\_dataset\_mo.py i inclou els passos següents:

1. **Generació de Processos per Tasca:**
   * Cada tasca conté un rang de processos assignats a processadors seleccionats aleatòriament. Aquest rang es pot modificar per observar com varia l’energia segons els processos que conten les tasques, tal com hem fet en el nostre treball.
   * Per cada procés es generen dades com: MI (Milions d'Instruccions), velocitat del processador (MIPS) i potència (kW).
2. **Generació de Deadlines:**
   * Els deadlines de les tasques es calculen a partir de la longitud mitjana de la tasca i la velocitat mitjana dels processadors, ajustats amb un factor aleatori entre 1.5 i 3. Serveix per ajustar els deadlines segons el nivell de dificultat de cada tasca. Això assegura que els terminis siguin adequadament exigents però realistes.
3. **Capacitat dels Processadors:**
   * Les capacitats es calculen com un factor proporcional a la velocitat del processador per simplificar i estandarditzar el model. Aquesta proporció permet capturar la relació directa entre la capacitat teòrica d'un processador i la seva velocitat en MIPS (Milions d'Instruccions Per Segon). Això és important perquè, a la pràctica, un processador més ràpid pot gestionar una major càrrega de treball.

Els datasets generats inclouen:

* process\_dataset.csv: Conté els processos associats a cada tasca i processador amb els seus paràmetres: identificador de la tasca, número de procés dins de la tasca, processador assignat, MI (longitud de la tasca en milions d’instruccions), velocitat del processador en milisegons i potència del procés en kilowatts.
* task\_deadlines.csv: Conté els deadlines de cada tasca en MI/velocitat.
* processor\_capacities.csv: Conté la capacitat de cada processador.

S’han generat diversos datasets variant el nombre de tasques, processadors i processos per tasca per analitzar com afecta aquestes variacions al consum d'energia. Aquest enfocament ha permès simular múltiples entorns realistes per testar el model implementat.

### **Implementació**

La implementació es realitza utilitzant Python i la llibreria pulp. El motiu és perquè tant la nostra funció objectiu com les restriccions són lineals. El codi es troba a l’arxiu optimitzacio.py i inclou els passos següents:

1. **Carregar els datasets:** Els datasets generats prèviament es carreguen i estructuren adequadament per facilitar la creació del model.
2. **Formulació de la funció objectiu:** Es modela la funció matemàtica que calcula el consum total d’energia en funció de les tasques assignades als processadors. El modelatge matemàtic i el procés d’optimització fan servir un solver (en aquest cas, pulp) per trobar la millor assignació de tasques als processadors que redueix al mínim el consum energètic. Pulp és una llibreria per a la programació lineal i optimització. Aquesta permet definir i resoldre problemes d’optimització mitjançant la creació de models matemàtics. Concretament, fem servir LpProblem("Minimitzar\_Energia", LpMinimize).
3. **Definició de les variables de decisió:** Es defineix una variable binària (xij) per a cada combinació de tasca i processador, indicant si la tasca s’assigna o no a un processador específic. Per fer-ho, hem declarat LpVariable( xij, cat="Binary").

La definició de variables de decisió es fa en aquesta etapa perquè les assignacions de tasques a processadors són un component central del model d’optimització i poden canviar dinàmicament durant el procés de resolució. A diferència de les dades fixes generades en el dataset (com MI, velocitats, potències, etc.), aquestes variables representen decisions que el model ha de prendre per trobar la solució òptima. Per tant, no es poden definir de forma estàtica durant la generació del dataset, sinó que es defineixen dins del procés d’optimització. Això permet que el solver consideri diferents combinacions d’assignacions per identificar la més eficient en termes de consum energètic.

1. **Aplicació de les restriccions:** Es codifiquen les limitacions de temps d’execució, capacitat dels processadors i assignació única de tasques. Es va servir LpSum() degut a que les restriccions són sumatoris: per cada tasca i cada processador.
2. **Resolució del problema:** Utilitzant el solver de pulp, es resol el model d’optimització per trobar la solució que minimitza el consum energètic: problem.solve(); print("Estat de la solució:", problem.status).

## 

## **Resultats i Discrepàncies**

### **Resultats de la Implementació**

Després de resoldre el model, els resultats mostren 3 diferents valors, degut a les seves respectives proves.

En la primera prova, es va plantejar un problema amb **25 tasques i 9 processadors**. Tot i que el model va trobar una solució amb un consum energètic de **20.76 kWh**, es va observar que aquesta no era factible, ja que algunes variables fraccionàries van sorgir a causa de restriccions ajustades en els deadlines. Aquest fet evidencia la necessitat de revisar els paràmetres per garantir solucions completament enteres. A més, el resultat no corresponia a un mínim global, sinó al pas on l'algoritme va determinar que no podia arribar a una solució globalment òptima.

En la segona prova, amb **50 tasques i 10 processadors**, el model també va trobar una solució amb un consum energètic de **31.52 kWh**. Tanmateix, aquesta configuració tampoc no era factible, per la qual cosa el resultat tampoc no representa un mínim global. Aquesta prova va permetre entendre que cal ajustar les restriccions per evitar solucions parcials o no enteres.

La tercera prova es va centrar en un escenari de gran escala amb **500 tasques i 200 processadors**. En aquest cas, el model va obtenir una solució òptima i factible amb un consum energètic mínim de 109.28 kWh. Això evidencia la capacitat del model per gestionar problemes més grans, mantenint solucions globalment òptimes, encara que a costa d’un temps de càlcul més elevat.

**Discrepàncies**

Les solucions òptimes de les dues primeres proves no complien totes les restriccions, resultant en solucions parcials no factibles. La causa principal d’aquestes discrepàncies rau en la generació de dades sintètiques, que no sempre reflecteix amb precisió les condicions reals. Això pot provocar inconsistències en les restriccions definides al problema i, per tant, en els resultats obtinguts. En concret:

1. **Primera prova (25 tasques i 9 processadors):** Les restriccions de capacitat i deadlines es van ajustar massa, fent que el model no trobés una solució completament factible, malgrat que el consum energètic calculat fos de 20.76 kWh.
2. **Segona prova (50 tasques i 10 processadors):** Les variables fraccionàries observades són una conseqüència directa d’aquestes discrepàncies en les dades, conduint a una solució parcial amb un consum energètic de 31.52 kWh, però no factible.

Aquestes discrepàncies assenyalen la necessitat de millorar el procés de generació de dades i revisar els paràmetres per assegurar que les condicions imposades siguin coherents i reflecteixin millor la realitat.

**Experimentació variables del model**

Durant el procés, hem experimentat amb diverses variables del model per entendre quines són clau per aconseguir solucions factibles i òptimes. Els aspectes més rellevants són els processos assignats a les tasques i els processadors:

1. **Assignació de tasques a processadors:** S'ha identificat que les tasques amb processos equivalents o superiors a la capacitat o velocitat del processador no es poden assignar. Si aquestes tasques s’assignen, no es compliran les restriccions establertes, i el model no serà factible.
2. **Revisió de paràmetres:** Això implica descartar automàticament aquestes assignacions invàlides durant l'execució del model per reduir inconsistències i garantir solucions que compleixin totes les restriccions.

Aquest ajustament en les variables del model és essencial per assegurar que les solucions obtingudes no només siguin òptimes, sinó també factibles dins del context definit pel problema.

## **Propostes de millora**

Per millorar la implementació ens agradaria:

* **Optimitzar el codi:** Utilitzar estructures de dades més eficients per gestionar els datasets.
* **Paràmetres dinàmics:** Ajustar les capacitats o velocitats dels processadors per explorar escenaris alternatius.
* **Escalabilitat:** Implementar solvers més avançats per manejar problemes amb molts processadors i tasques.

## 

## **Conclusió i Treball Futur**

Aquest projecte ha demostrat com els models matemàtics i les eines d'optimització poden ser utilitzats per minimitzar el consum energètic en l’assignació de tasques a processadors. Els resultats obtinguts són satisfactoris i confirmen la validesa del model implementat.

Les proves realitzades demostren que el model és robust i capaç de gestionar diferents escenaris, des de configuracions petites fins a sistemes de gran escala.

**Treball Futur**

* **Ampliació del model:** Considerar paràmetres addicionals com la latència de comunicació entre processadors.
* **Treballar amb dades del món real:** Trobar un dataset que s’ajusti al nostre problema i treballar a partir d’aquelles dades per no haver de generar dades sintètiques.
* **Exploració de mètodes heurístics:** Per a problemes de major escala on els solvers exactes poden ser ineficients.