Rapport de Java

« Occupancy Estimator »

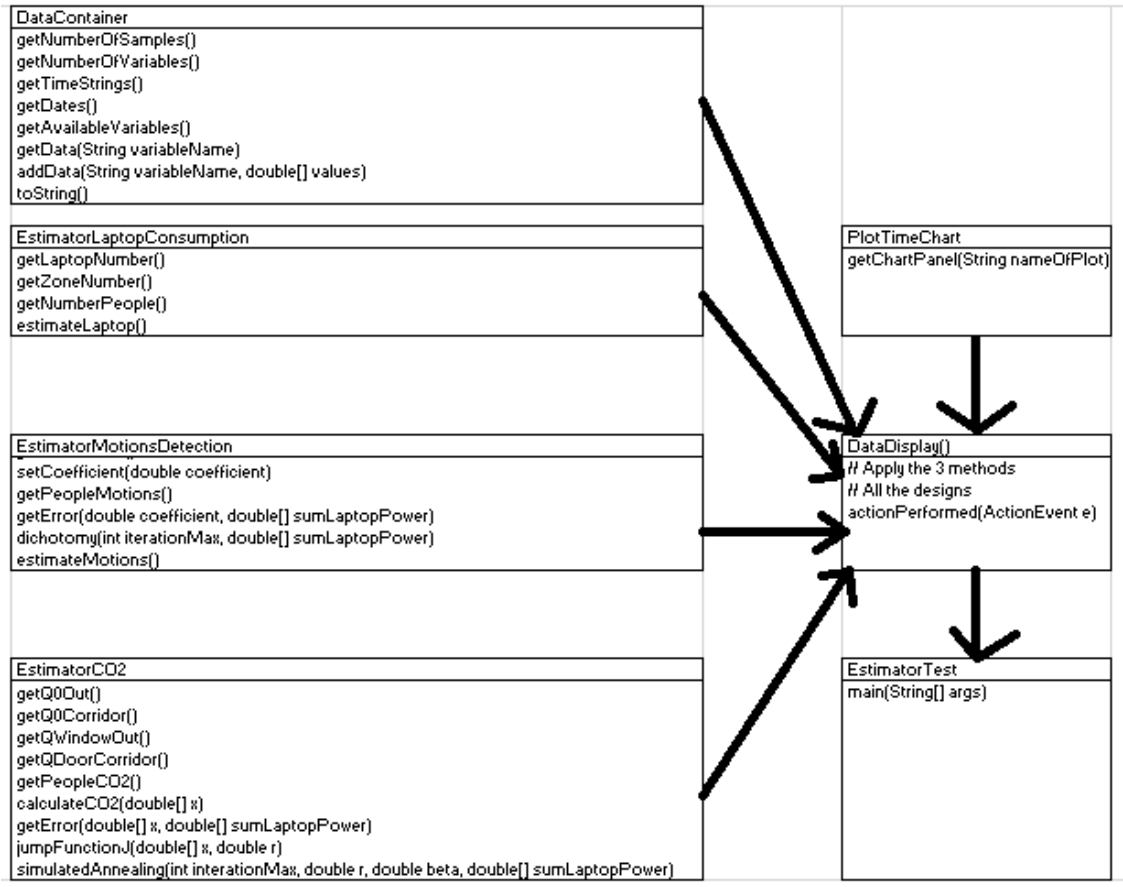
Étudiant: TRAN Gia Quoc Bao, groupe G4-c, ASI Date: 16 Janvier 2020

Professeurs: MEUNIER Gerard & LARANJEIRA Tiansi

Introduction au problème:

L'objectif de ce projet est d'estimer le nombre de personnes présentes dans une salle de bureau, à l'aide d'une programmation orientée objet et des données collectées concernant la consommation électrique des ordinateurs portables, le nombre de mouvements détectés et la quantité de CO_2 libérée. Ici, j'ai utilisé le langage Java pour tracer d'abord les données, puis concevoir 3 estimateurs d'occupation différents. Chaque classe doit avoir moins de 200 lignes et être orientée objet. De plus, les conventions de nommage Java selon Oracle étaient bien respectées. Enfin, j'ai analysé les méthodes et commenté leur efficacité.

Diagramme UML du code Java que j'ai utilisé:



L'explication de chaque question:

1. Concevez une interface graphique pour générer des tracés des variables sélectionnées:

J'ai conçu une interface graphique avec des cases à cocher qui correspondent aux variables et un bouton «plot» (correspondant à toutes les données disponibles et aux 3 données ajoutées ultérieurement). Chaque fois on presse le bouton, une boucle vérifierait quelles données sont sélectionnées et toutes sont tracées dans la même fenêtre.

Les noms des données tracées sont également affichés: si aucune variable n'est choisie, l'axe « y » et le titre affichent «Measurements» mais si vous choisissez certaines variables, elles afficheront leurs noms. Plus précisément, si nous sélectionnons 1 variable, son nom est affiché et si nous sélectionnons plus de 1, tous les noms sont affichés séparés par «&». Bien entendu, lorsque nous modifions les données sélectionnées, le noms changent également.

J'ai également ajouté une image à des fins d'information et de décoration. Je ne me concentre pas trop sur la beauté de l'interface graphique car je pense que c'est le travail des designers et l'interface graphique doit pour ma part être claire et facile à utiliser, les algorithmes sont ce qui m'intéresse particulièrement.

2. Estimateur d'occupation basé sur la consommation d'énergie des ordinateurs portables:

J'ai créé 4 objets avec ces 3 variables: le numéro d'ordinateur portable, le numéro de zone et le nom du fichier à partir duquel les données sont importées. Chacun des objets contient une ArrayList où nous écrirons 1 ou 0 - quelqu'un utilise l'ordinateur portable à ce moment-là ou non. Une méthode nommée «estimationLaptop» vérifiera si la consommation d'énergie à ce moment est supérieure à la valeur seuil de 15 W ou non et si oui, elle ajouterait 1 à la liste de tableaux. Enfin, j'ai ajouté les résultats (dans le « sumLaptopPower ») pour obtenir le nombre total de personnes et l'ai tracé. Le « sumLaptopPower » sera utilisé plus tard pour régler les 2 autres méthodes. Les méthodes à l'intérieur de cette classe:

• Les méthodes de Getters, car les données doivent être cachées dans «privé».

• Une méthode appelée «estimationLaptop» fait les travaux comme décrit.

3. Estimateur d'occupation basé sur la détection de mouvement:

Étant donné que le nombre de personnes est proportionnel au nombre de mouvements détectés, le coefficient est déterminé en minimisant l'erreur entre cet estimateur et le précédent, en utilisant l'algorithme de dichotomie. La fonction d'erreur que nous souhaitons minimiser est:

$$f(C) = \sum |people\_motion(C) \; - \; peopleLaptop| = \sum |C \times motionsDetected - occupancyLaptop|$$

J'ai utilisé la valeur absolue car les erreurs mathématiques peuvent être positives ou négatives et l'utilisation de valeurs absolues est plus précise. De plus, je n'ai pas arrondi les résultats: vous verrez que les résultats ont exactement la même forme que les «émotions_détectées» d'Excel, mais avec des valeurs plus petites, car nous avons ajusté le coefficient en utilisant le « sumLaptopPower » obtenu précédemment.

Les méthodes à l'intérieur de cette classe:

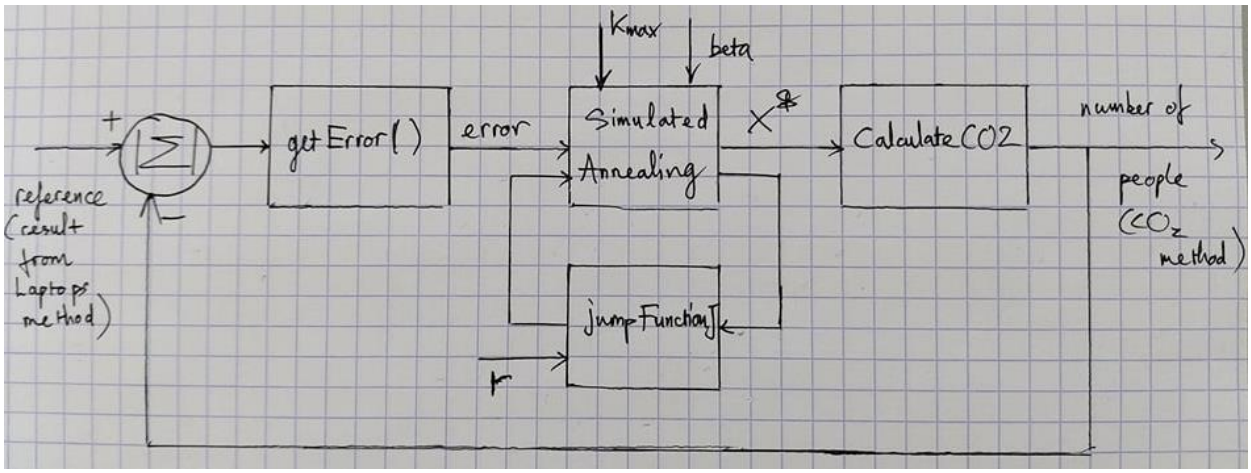
• Les méthodes de Getters, car les données doivent être cachées dans «privé».

• Une méthode appelée «dichotomy» pour utiliser l'algorithme de dichotomie, qui renvoie un vecteur dont le 1er élément est le coefficient accordé et le 2e est la précision. Je peux mettre la dichotomie dans la fonction principale car cette fois, nous n'avons besoin de le faire qu'une fois, mais j'en ai fait une méthode car elle est plus générale et peut être réutilisée si nous avons plus de données.

• Une méthode appelée «estimationMotions» parcourra les données et estimera le nombre de personnes, tout comme celle basée sur ordinateur portable.

4. Estimateur d'occupation basé sur la concentration de CO_2:

J'ai choisi de faire cette partie car je m'intéresse aux algorithmes et cela devrait aider à améliorer mes compétences mathématiques ainsi que mes compétences en programmation. J'ai besoin de régler les 4 coefficients en utilisant « Simulated Annealing ». Il peut y avoir différentes façons de mettre en œuvre l'algorithme mais en tant qu'étudiant en contrôle automatique, je préfère cette façon de l'exprimer. Appelons X, qui a une taille de 4 éléments, le vecteur que nous voulons optimiser. Ensuite, l'algorithme que j'utilise est décrit en utilisant ce diagramme:



J'ai donc utilisé 4 méthodes (illustrées dans le diagramme) pour obtenir un bon ensemble de coefficients et obtenir la valeur estimée du nombre de personnes.

Le seul problème que j'ai eu avec l'utilisation de cet algorithme est qu'ici, comme personne ne sait quelle est la fonction f' dans l'algorithme, je devais supposer qu'elle est égale à $f(X^r)$. Ce n'est probablement pas correct, comme nous le voyons à la 9ème ligne de la description de l'algorithme dans le livre qu'ils ont écrit $f(X^r)$ et f', donc ce n'est pas la même chose. C'est peut-être la raison pour laquelle j'ai obtenu des valeurs négatives pour les résultats.

Les méthodes à l'intérieur de cette classe (ce qui m'a vraiment intéressé):

• Les méthodes de Getters, car les données doivent être cachées dans «privé».

• Une méthode appelée «CalculateCO2» calculera le nombre de personnes lorsque nous lui donnerons le vecteur X des 4 coefficients.

• Une méthode appelée «getError» obtient l'erreur entre la méthode basée sur CO_2 et la méthode basée sur ordinateur portable.

• Une méthode appelée «jumpFunctionJ» effectue le saut aléatoire lorsque nous lui donnons un vecteur et un rayon.

• Une méthode appelée «simulatedAnnealing» exécute l'algorithme et nous donne les coefficients accordés.

5. Analyse des résultats et conclusion:

J'ai comparé les méthodes que nous avons utilisées pour estimer l'occupation:

Estimateur	Points forts	Points faibles
Consommation d'énergie des ordinateurs portables	Le moins cher.	Imprécision: peut-être 2 personnes ou plus utilisent le même ordinateur portable; peut-être que les gens sont dans la pièce mais n'utilisent pas d'ordinateurs portables; peut-être que l'ordinateur portable fonctionne mais que l'utilisateur n'est pas là; ... Le nombre de personnes est limité à 4 car nous ne faisons pas de distinction entre 1 utilisateur et 2 utilisateurs du même ordinateur portable.
Détection de mouvement	Prix pas trop élevé.	La précision de celui-ci est basée sur celle de la première méthode (limitée), car nous avons essayé d'obtenir le coefficient en minimisant l'erreur entre les deux. Imprécision: peut-être que les gens sont dans la pièce mais qu'ils bougent à peine; peut-être que les gens ouvrent la porte juste pour regarder au lieu d'aller à l'intérieur ou à l'extérieur; peut-être que la porte s'est ouverte une fois mais beaucoup de gens entrent dans la pièce; ...
Concentration de CO_2	La méthode la plus précise car les gens respirent toujours et la quantité de CO_2 libérée est très proportionnelle au nombre de personnes, car tous les adultes ont tendance à produire la même quantité de CO_2.	Imprécision: peut-être que 2 bébés produisent la même quantité de CO_2 qu'un adulte, ou qu'il peut y avoir des animaux, mais ce sont rarement le cas; ... Le plus cher des trois. En outre, il n'est peut-être pas raisonnable de dépenser beaucoup d'argent pour équiper des capteurs de CO_2 juste pour savoir combien de personnes se trouvent dans la pièce. Nous pouvons avoir d'autres méthodes à la fois efficaces et bon marché, comme l'utilisation de «cartes d'identité»: les employés scannent leurs cartes chaque fois qu'ils entrent dans la pièce.

Conclusion:

La méthode la plus précise ici est la dernière, car elle est difficile à construire et à programmer et a un coût élevé. Donc, la méthode 1 ou 2 suffit si nous voulons simplement estimer le nombre de personnes.

Dans ce projet, j'ai pu programmer 3 estimateurs différents, même si le dernier a donné des résultats étranges car l'explication sur le f^{r} n'était pas tout à fait claire.

L'un des avantages de la programmation orientée objet est qu'il est très pratique de manipuler des objets du même type et d'étendre davantage nos données. Par exemple, si nous avons plus de zones avec plus d'ordinateurs portables, nous pouvons simplement ajouter 1 ou 2 lignes pour mettre à jour nos codes sans avoir à tout réécrire.

Merci