MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE «ABBES LAGHROUR» DE KHENCHELA
FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département de Math et Informatique

N° de série :……

Mémoire de fin d'études

Pour l'obtention du diplôme de Master (L.M.D)

Spécialité : sécurité et technologie web
Option : sécurité et technologie web

Approche d'optimisation d'énergie dans le mobile cloud computing

Réalisé par : - Bohalel Selma
- Djeddi Ahlem

Dirigé par : dr.hadjer faleh

Session 2022

1

Dédicace

À mes parents, mes frère .

Salma bouhlal

À ma mère d'avoir été patiente avec moi , à mon frère et à mon neveu .

Ahlam djedi

Remerciements

Je tiens d'abord à remercier mon encadreur, le Dr fellah hadjer, pour la qualité de son encadrement. Sa disponibilité et son attention au début et la fin de ce mémoire, ont été pour moi une aide précieuse pour la réalisation de ce travail. Mes remerciements s'adressent également aux honorables membres du jury qui ont acceptés d'évaluer mon travail.

ملخص

أصبحت الأجهزة المحمولة الذكية، مثل الهواتف الذكية وأجهزة الكمبيوتر اللوحية ، أداة أساسية في حياة الإنسان في الوقت الحاضر. بالنسبة لمستخدمي جميع الأجهزة المحمولة ، تعد الهواتف الذكية هي الأدوات الأكثر ملاءمة والأكثر أهمية والتي لا تستخدم فقط للاتصال. ولكن أيضًا لأغراض أخرى مثل الألعاب. و العمل. علاوة على ذلك ، فإن الهواتف الذكية خالية تمامًا من حدود الزمان والمكان. على سبيل المثال ، في سيناريو الطوارئ ، يمكن للمستخدم تعديل أعماله وتحسينها بسرعة باستخدام Google Drive. على الهاتف Galaxy Note عندما يكون في مترو الأنفاق. إلى جانب ذلك ، يتوقع مستخدمو الأجهزة المحمولة دائمًا إمكانية دعم نفس التطبيقات على أجهزة الهاتف الخاصة بهم عندما يتنقلون بدلاً من إحضار أجهزة الكمبيوتر المحمولة الأثقل نسبيًا أو الجلوس أمام أجهزة كمبيوتر سطح مكتب قوية غير قابلة للحركة. يعد تنفيذ تطبيقات الهاتف المحمول باستخدام موارد السحابة الحسابية أحدث استراتيجية لزيادة قيود الموارد على الأجهزة المحمولة الذكية. يتطلب التحميل الحسابي تقسيم تطبيق الهاتف المحمول أثناء تنفيذ التطبيق. نظرًا لأن نهج التقسيم الأمثل يعد الأفضل لتحسين و توفير الطاقة والأداء على الهواتف الذكية، لذلك فإن تقسيم تطبيقات الهاتف المحمول في وقت التشغيل يعد منظورًا بحثيًا صعبًا.

الكلمات المفتاحية: الحوسبة السحابية، تحميل التطبيقات الحسابية، تجزئة التطبيق، تحسين استهلاك الطاقة.

Abstract

Smart Mobile devices (SMDs), such as smartphone and tablet PCs, are becoming an essential tool of human life nowadays. For mobile users, SMDs is the most convenient and essential tools that not only used for communication. but also for other purposes such as gaming. and working. Furthermore, SMDs are totally free from the boundaries of time and place. For example, in the scenario of emergency, an user can quickly edit and improve his works with Google Drive. using Galaxy Note when he is in the subway. Besides, mobile users always have high expectation that the same applications can be supported on their SMDs when they are moving around rather than bringing the relatively heavier laptops or sitting in front of an unmoveable powerful desktop PCs. The execution of mobile application using computational cloud resources is the latest augmentation strategy for resources constraint Smart Mobile Devices (SMDs). Computational offloading requires mobile application to be partitioned during the execution of the application on SMDs. Since an optimal partitioning approach promises optimization of energy savings and performance on SMDs, partitioning of mobile application at runtime is a challenging research perspective.

Keys words: Mobile Cloud Computing, Computation Offloading, Application Partitioning , Energy optimization,

Résumé

Les appareils mobiles intelligents (SMD), tels que les smartphones et les tablettes, deviennent de nos jours un outil essentiel de la vie humaine. Pour les utilisateurs mobiles, les SMD sont les outils les plus pratiques qui ne sont pas seulement utilisés pour la communication. mais aussi à d'autres choses telles que les jeux et le travail. De plus, les SMD sont totalement libérés des limites de temps et de lieu. Par exemple, dans le scénario d'urgence, un utilisateur peut rapidement modifier et améliorer ses travaux avec Google Drive. utilisant Galaxy Note lorsqu'il est dans le métro. En outre, les utilisateurs mobiles s'attendent toujours à ce que les mêmes applications puissent être prises en charge sur leurs SMD lorsqu'ils se déplacent plutôt que d'apporter des ordinateurs portables relativement plus lourds ou de s'asseoir devant un ordinateur de bureau puissant et immobile. L'exécution d'applications mobiles à l'aide de ressources cloud de calcul est la dernière stratégie d'augmentation des contraintes de ressources des Smart Mobile Devices (SMD). Le déchargement informatique nécessite que l'application mobile soit partitionnée lors de l'exécution de l'application sur les SMD. Puisqu'une approche de partitionnement optimale promet une optimisation des économies d'énergie et des performances sur les CMS, le partitionnement de l'application mobile au moment de l'exécution est une perspective de recherche difficile.

Mots Clés : L'informatique Mobile en nuage, Déchargement de calculs, Partitionnement d'applications, Optimisation d'énergie.

Table des matières

Listes des figures

Listes des Tableaux

1 Introduction Générale

1.1 Contexte du projet :

Aujourd'hui les appareils mobiles notamment les téléphones intelligents et les tablettes sont devenus très nombreux et font partie de notre vie quotidienne. La popularité des appareils mobiles augmente chaque jour, en 2019 le nombre de dispositif mobile dépasse les cinq milliards de dispositif. Cette inflation a permis l'amélioration des appareils mobiles dans divers aspects tels que le CPU mémoire stockage, la taille de l'écran etc. Cela dit ils ont encore des limitations parmi lesquelles nous citons les ressources énergétiques de calcul intensif et déploiement d'applications complexes. Ce manque de ressource constitue un obstacle, pour le traiter ils ont utilisé les ressource partagées, d'un autre coté une technologie apparait c'est l'informatique en nuage (en anglais cloud computing) qui désigne un ensemble de processus qui consiste à utiliser la puissance de calcul et de stockage de serveurs informatiques distants à travers un réseau permettant aux individus et aux entreprises d'utiliser des logiciels ou des matériels qui sont gérés par les fournisseurs sur le site distant. Le model de cloud permet d'accéder à des sources d'information et à des ressources informatiques depuis n'importe quel endroit. De la fusion de cloud avec le mobile est née le cloud mobile qui répond aux limites des appareils mobiles. Le MCC est une technologie qui intègre le cloud computing aux dispositifs mobiles, il permet l'utilisation et l'accès aux services de nuage à l'aide de dispositifs mobiles. Parmi les ressources importantes : des applications tels que le traitement d'image, la reconnaissance vocale et faciale et la traduction.

1.2 Problématique

Ces dernières applications mobiles ont besoins d'une grande puissance de calcul informatique de mémoire d'énergie et de bandes passantes, la capacité limitée des terminaux mobiles pose un problème qui oblige la communauté scientifique à créer des solutions. La déportation des taches de calcul dans le cloud a été proposée pour résoudre le problème de performance des dispositifs mobiles et réduire la consommation d'énergie cette approche n'est pas toujours la solution idéale. Plusieurs propositions sont faites les unes plus performantes que les autres. L'objectif de cette recherche est donc d'apporter des solutions aux problèmes de performance. Afin de trouver une réponse aux problèmes, nous répondrons aux questions suivantes : quand est-ce que la déportation des taches dans le cloud doit être

faite? Comment identifier les différentes parties du code à déporter? Comment optimiser la consommation de différentes ressources?

1.3 Solutions proposées

L'objectif de toutes les approches proposées est d'optimiser la consommation des ressources

De façon générale, ce mémoire vise à :

- ✓ Etudier le domaine du MCC.
- ✓ Analyser les approches et algorithmes d'optimisations existants dans ce domaine.
- ✓ Proposée une approche d'optimisation de la consommation d'énergie.
- ✓ Evaluer les performances de l'approche proposée.

Ainsi, l'objectif principal de ce travail est de proposer une approche d'optimisation de la consommation d'énergie pour la déportation de calculs d'applications dans le Mobile Cloud Computing. Pour ce faire, nous nous inspirons de travaux connexes récents tels que MAUI et CLONECLOUD et aussi de l'architecture de l'informatique organique centralisée.
De façon spécifique, Nous cherchons a trouvé l'approche qui répond à cet ensemble de critères :

- ✓ Minimiser la consommation d'énergie et prédire l'exécution pour la prise de décision.
- ✓ Avoir une approche consciente de son environnement, autonome et cognitif (apprentissage à base de cas).
- ✓ Trouver le partitionnement optimal dans un état N.

1.4 Organisation du manuscrit

Ce manuscrit est divisé en deux grandes parties : une première partie consacrée à la présentation de l'état de l'art, elle a pour but de présenter une synthèse des différents domaines auxquels nous avons touché dans le cadre de ce travail. La deuxième partie est consacrée à notre contribution.

ETAT DE L'ART

Chapitre 1 : divisé en 3 sections, il introduit les concepts Cloud Computing, Mobile Computing et le Mobile Cloud Computing.

Chapitre 2 : présente le mécanisme de déportation de calcul et les ressources consommatrices d'énergie.

CONTRIBUTION

Chapitre 3 : nous présenterons notre approche proposée avec ses différents modules internes et leurs interactions et rôle de chacun.

Chapitre 4 : nous commencerons par introduire l'environnement de développement utilisé et la plateforme choisie pour la réalisation de nos modules, Enfin nous choisirons une application énergivore en commentant les différents résultats obtenus par le scénario.

1 Chapitre 1 : Mobile Cloud Computing et Déchargement de calculs

1.1 Introduction :

Le cloud computing mobile est très célèbre en raison de sa manière de minimiser la consommation d'énergie et d'améliorer l'expérience des utilisateurs en externalisant les applications/opérations gourmandes en ressources/calcul des appareils mobiles vers les clouds. Le problème majeur des appareils mobiles est la faible autonomie de la batterie, les ressources et la capacité de stockage limitées. Pour surmonter ces contraintes, le cloud computing mobile a introduit le déchargement, la manière dont le stockage et les calculs de données sont effectués à l'intérieur d'un cloud distant plutôt que d'un processeur et d'un stockage mobiles. Mais en utilisant le déchargement, le cloud computing mobile nécessitait un réseau mobile économe en énergie et un environnement cloud. Le stockage et la sécurité sont également devenus des enjeux importants.

1.1 Mobile Cloud Computing :

1.1.1 Cloud computing

Nous considérons la définition de Cloud Computing proposée par l'Institut national de la norme et de la technologie (NIST), car elle illustre les aspects essentiels du Cloud : « Le Cloud Computing est un modèle permettant un accès aisé, à la demande et au travers d'un réseau, à un ensemble partagé de ressources informatiques (par exemple des serveurs, des espaces de stockage, des applications) qui peuvent être rapidement mises en service avec un effort minimum de gestion et d'interaction avec le fournisseur de ce service. »

Avec le Cloud Computing, Les machines, les applications et les données pourront être disséminées ou centralisées dans un, ou dans différents sites internes, chez des prestataires, dans un data center « datacenters » situé à l'autre bout de la planète ou sur une myriade de serveurs appartenant à un même « nuage »(Mell)

Un nuage (anglais cloud) est un ensemble de matériel, de raccordements réseau et de logiciels qui fournit des services sophistiqués que les individus et les collectivités peuvent exploiter à volonté depuis n'importe où dans le monde

Le Cloud Computing est un modèle basé sur Internet qui fournit un accès à la demande au réseau qui consiste en des ressources informatiques partagées (par exemple, des réseaux, serveurs, stockage, applications et services), un effort de gestion minimal Le nuage organise un grand nombre de ressources (calcul, réseau, et stockage) en fonction d'un plan de gestion particulier. Il fournit des applications et du matériel sous forme de services qui sont disponibles partout et à tout moment.(vi, m, y, & allan , 2009/2010)

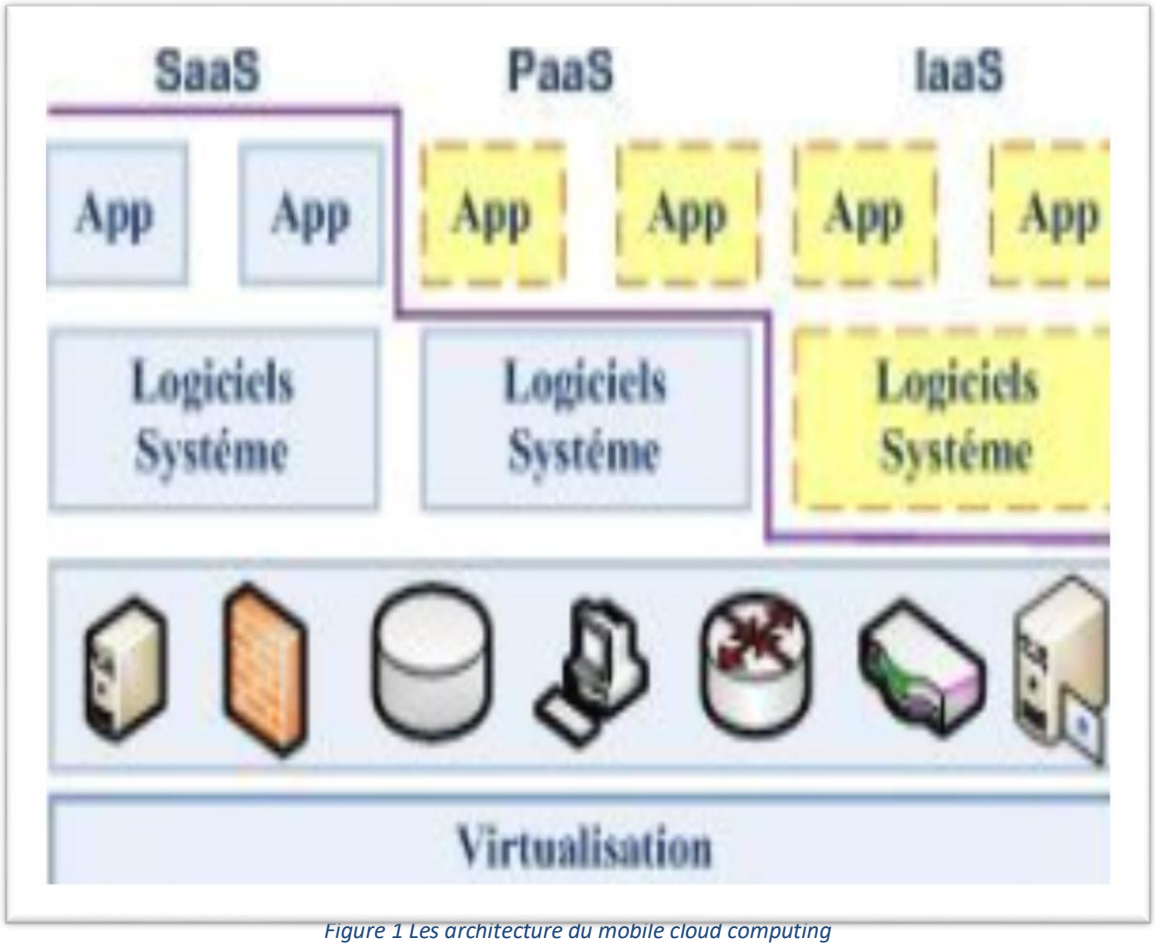


Figure 1 Les architecture du mobile cloud computing

1.1.2 Mobile cloud computing MCC

Les appareils mobiles ont des capacités et des performances limitées ,ces limitations existent dans de nombreux domaines, notamment la puissance de traitement, la capacité

énergétique ainsi que les restrictions liées à la taille et au facteur de forme les appareils mobiles ont tendance à être plus lents à traiter les informations, à fonctionner lentement et avec des performances de connectivité similaires ou plus lentes.. Les nuages mobiles ont le potentiel de combattre à nouveau les limitations des appareils mobiles, L'un des défis fondamentaux des communications mobiles est le fait que tous les appareils mobiles sont des systèmes portables à énergie limitée, car ils fonctionnent sur batterie. Les progrès de la technologie des batteries, en particulier ceux visant à augmenter radicalement la capacité énergétique par unité de volume,. La disponibilité d'une quantité limitée d'énergie à bord d'une part, avec la tendance actuelle à intégrer de plus en plus de fonctionnalités gourmandes en énergie d'autre part, constituent l'un des défis les plus difficiles auxquels sont confrontés les fabricants d'appareils mobiles aujourd'hui. Certes, les limitations énergétiques créent également de nombreuses restrictions et caractéristiques indésirables pour les utilisateurs, telles que des durées de fonctionnement courtes et la nécessité de recharger fréquemment la batterie

1.1.3 **Architecture Informatique de MCC:**

Nous examinons d'abord trois architectures cloud nouvellement proposées et représentatives pour l'informatique mobile. Ces architectures sont conçues pour utiliser les nouvelles fonctionnalités des appareils mobiles et répondre aux nouveaux besoins des applications mobiles. (Q & M, s.d.)

1.1.3.1 Architectures client-serveur mobiles :

Dans ce modèle, les téléphones mobiles fonctionnent comme des clients légers dans le sens où ils sont similaires à un ordinateur client ordinaire qui ne fournit qu'une interface utilisateur **(UI)** pour parcourir un serveur, et les serveurs cloud exécutent toutes les applications. Le téléphone mobile demande l'exécution d'applications à forte intensité de calcul à partir du cloud d'une manière similaire à la demande de ressources à partir de serveurs Web. Cependant, ce modèle client-serveur traditionnel ne tire aucun avantage des smartphones et néglige bon nombre des fonctionnalités nouvelles et uniques qu'ils possèdent.

1.1.3.2 Architectures de cloudlets :

Bien que l'architecture cloud client-serveur puisse prendre en charge les besoins de l'informatique mobile, elle est affectée par la limitation inhérente des liens d'accès des appareils mobiles. La limitation fondamentale est la latence dans le réseau. Dans le cloud, les

appareils mobiles interagissent avec les serveurs cloud sur le réseau étendu sans fil (WWAN) et Internet. Les deux réseaux subissent un retard aller-retour et une gigue non négligeable pour les applications mobiles interactives. Un tel retard et une telle gigue ont en grande partie un impact négatif sur la perception et l'engagement cognitif de l'utilisateur. Une autre latence.

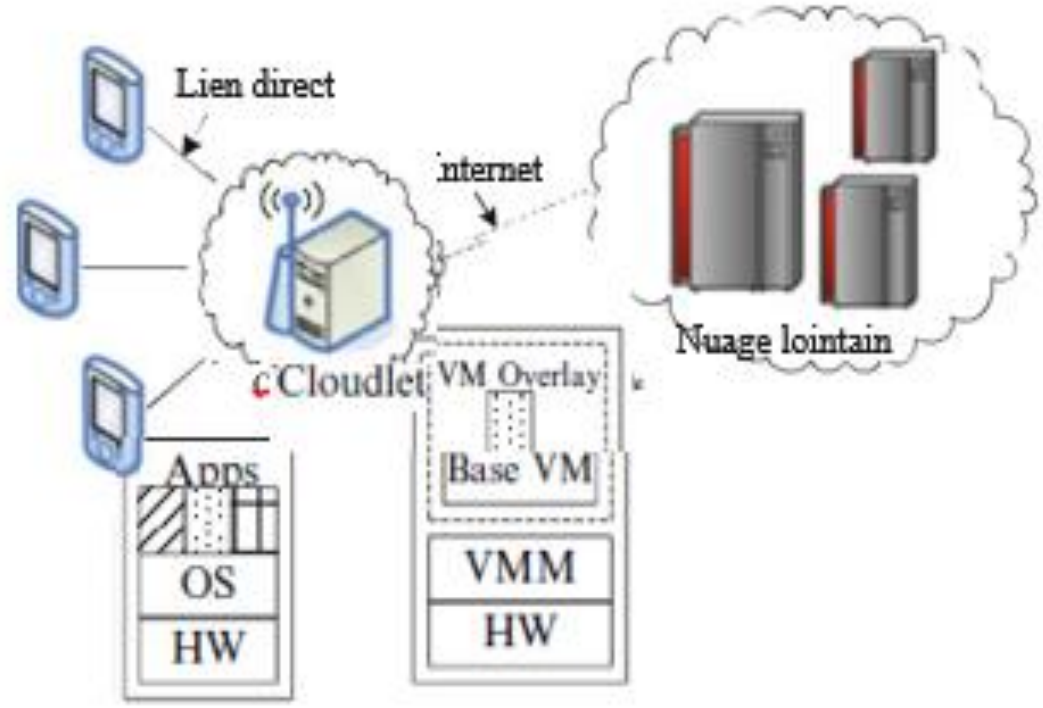


Figure 2 Architecture Cloudlet

1.1.3.3 Architectures cloud mobiles ad hoc:

Bien que les deux premières architectures cloud pour l'informatique mobile soient prometteuses dans la gestion et l'exécution d'applications pour appareils mobiles, elles ont besoin d'une infrastructure pour héberger le service cloud et fournir un accès aux appareils mobiles. Il n'est pas rare que l'accès à une telle infrastructure ne soit pas disponible pour les utilisateurs mobiles dans diverses circonstances. Par exemple, lors d'une catastrophe naturelle (par exemple, un ouragan) qui détruit une partie de l'infrastructure, les appareils mobiles peuvent difficilement accéder à des services autres que se détecter eux-mêmes via Bluetooth et WiFi en mode ad hoc. Il peut également être trop coûteux d'accéder au service cloud via l'infrastructure, en particulier dans les zones où seules des liaisons de données sans fil à coût élevé (par exemple, une liaison cellulaire et une liaison satellite) sont disponibles. Étant donné que le volume de données en vrac et de code à décharger est bien supérieur à la quantité de données vocales, le coût de l'exécution du cloud computing via ces liaisons sans fil à coût élevé peut largement annuler les avantages obtenus des services cloud.

Une nouvelle solution cloud mobile ad hoc a été proposée danspour résoudre les problèmes soulevés lorsqu'aucun accès aux services cloud n'est disponible. La solution est particulièrement adaptée aux besoins des utilisateurs mobiles qui se trouvent dans des endroits proches et partagent des activités communes. Par exemple, lorsqu'un touriste visite des musées, il peut s'intéresser à la description des expositions. Il peut prendre des photos du texte et exécuter un logiciel de reconnaissance de texte pour le reconnaître afin de pouvoir stocker le texte sur son téléphone portable. Cependant, le traitement de l'intégralité du texte nécessite plus de ressources informatiques que ses capacités téléphoniques.

1.2 Déchargement de calculs

1.2.1 Défintion de déchargement :

Le concept de déchargement n'est pas un nouveau concept, il est présenté depuis les années 1970 avec des idées similaires à l'équilibrage de charge dans les systèmes distribués. Le cloud computing mobile (MCC) offre la possibilité d'économiser de l'énergie en effectuant des calculs lourds à distance sur le cloud.. La figure (1) décrit le concept de déchargement des composants de l'application à exécuter sur le cloud en tant que tâches parallèles..(K, tawalbej, & m)

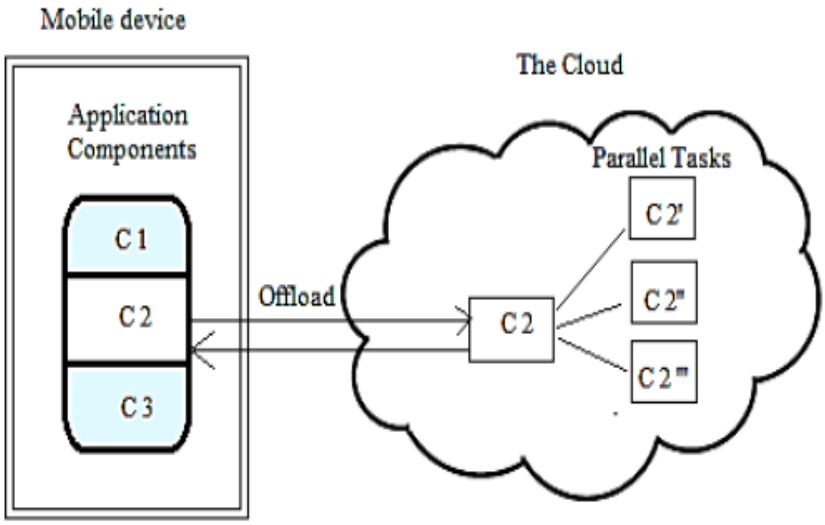


Figure 3 Concept de déchargement

1.2.2 Le déchargement à deux niveaux

Dans le déchargement à deux niveaux, seules deux entités sont impliquées, l'appareil mobile et le cloud. L'appareil mobile demande au cloud d'exécuter cette application dessus. Un appareil mobile peut décharger une partie de sa charge de travail sur un puissant serveur cloud via un ou plusieurs réseaux de communication, en tirant parti des ressources cloud abondantes pour collecter, stocker et traiter les données. Ce schéma de déchargement dépend de la disponibilité de cette ressource particulière sur le cloud.(Management Research Offloading Approach In Mobile Cloud Computing)

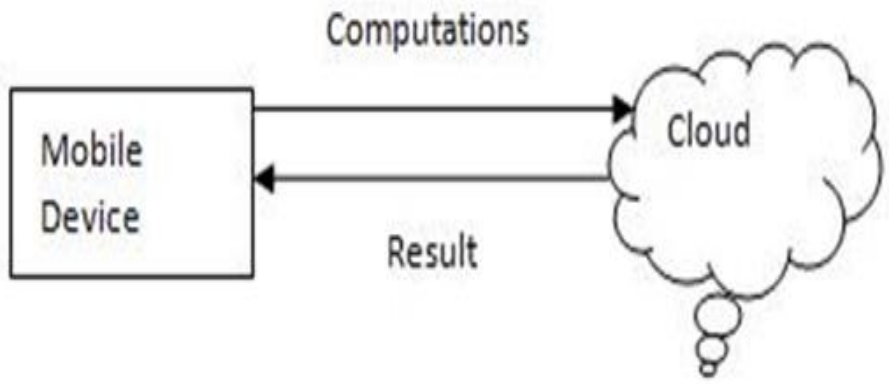


Figure 4 Le déchargement à deux niveaux

Le déchargement signifie le transfert de données d'un ordinateur ou d'un appareil numérique vers un autre appareil numérique. Est une solution pour augmenter les capacités des systèmes mobiles en migrant le calcul vers des ordinateurs plus ingénieux, tels que des serveurs. Ceci est différent de l'architecture client-serveur traditionnelle, alors que le client léger migre toujours le calcul vers un serveur. Le déchargement de calcul est également différent du modèle de migration utilisé dans les systèmes multiprocesseurs et l'informatique en grille, où un processus peut être migré pour l'équilibrage de charge. La principale différence est que le déchargement des calculs migre les programmes vers des serveurs en dehors de l'environnement informatique immédiat des utilisateurs ; la migration de processus pour l'informatique en grille se produit généralement d'un ordinateur à un autre dans le même environnement informatique.

1.2.3 Le déchargement à trois niveaux :

Dans le déchargement à trois niveaux, il existe une autre entité connue sous le nom de cloudlet. Un cloudlet est un centre de données ou un cluster d'ordinateurs à petite échelle conçu pour fournir rapidement des services de cloud computing aux appareils mobiles, tels que les smartphones, les tablettes et les appareils portables, à proximité géographique. Plutôt que de s'appuyer sur un cloud distant, la pauvreté des ressources d'un appareil mobile peut être résolue en utilisant un cloudlet riche en ressources à proximité via un point d'accès WLAN pour réduire la latence et réduire la consommation de la batterie.(Management Research Offloading Approach In Mobile Cloud Computing)

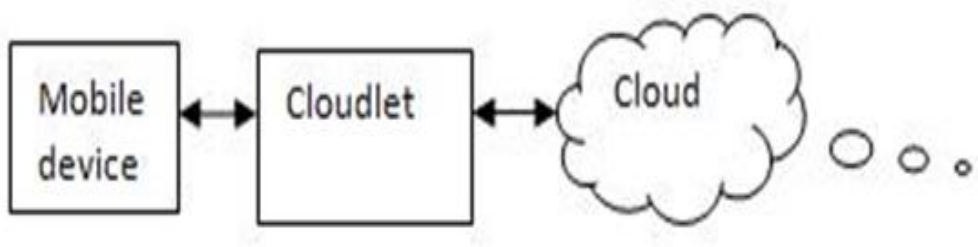


Figure 5 Le déchargement à trois niveaux

1.2.4 Architecture de déchargment

La figure Montre l'architecture générale du processus de déchargement. Lors du déchargement, les informations sont d'abord collectées sur l'appareil et les caractéristiques du réseau. Ces informations sont liées au coût de calcul et de communication de l'exécution, puis sur la base de ces informations, la décision de déchargement est prise comme quoi, quand, où et comment décharger l'application sur le cloud, une fois la décision prise, le processus de déchargement commence. Au cours de cette étape le type de déchargement est décidé statique ou dynamique puis le partitionnement de l'application est effectué. Le partitionnement peut être local ou distant, dans le partitionnement local, l'application est exécutée localement sur l'appareil mobile tandis que dans le partitionnement à distance, l'application est déchargée sur le serveur cloud en découvrant si un cloud est disponible ou non. Une fois l'application déchargée sur le serveur cloud, elle s'y exécute, puis le résultat est retransféré sur l'appareil mobile.

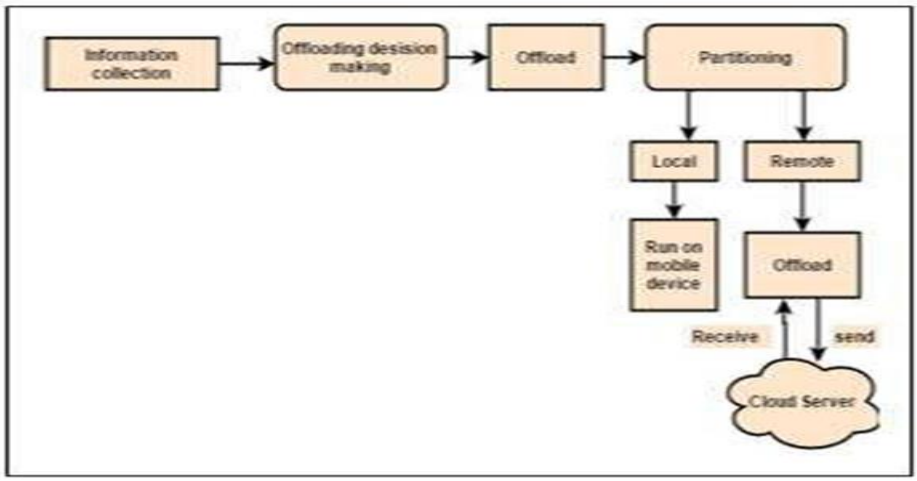


Figure 6 Architecture de déchargement

1.2.5 Topologies de déchargement :

Une connexion est illustrée entre les appareils mobiles et le cloud de déchargement multi-cloud autonome dans la figure7. C'est ce qu'on appelle la topologie en étoile. Si l'appareil mobile communique avec différents serveurs, beaucoup de temps et d'énergie sont consacrés à la communication avec plusieurs serveurs. La figure7 présente également une topologie en anneau où une connexion existe entre les appareils mobiles et le cloud de déchargement multicloud autonome. Ici, deux nœuds communiquent entre eux et dépendent d'autres nœuds. Lorsqu'un échec se produit au milieu des nœuds, le programme ne peut pas être exécuté avec succès.

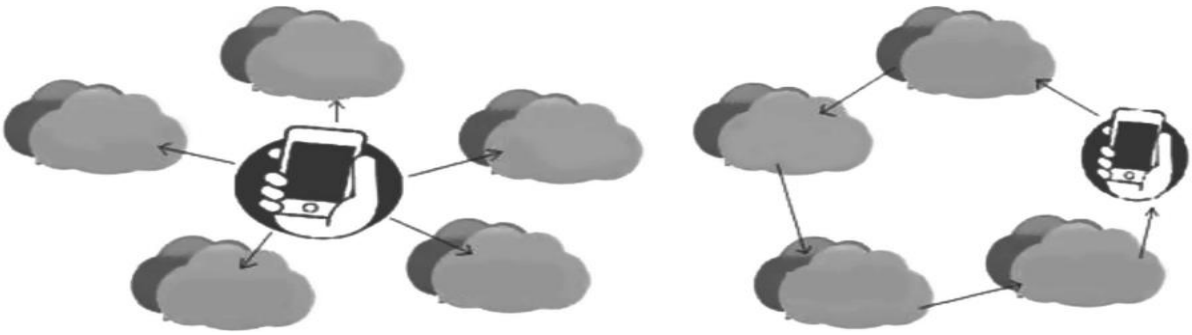


Figure 7 (a)Topologie en étoile de déchargement MCC et (b) Topologie en anneau de déchargement MCC

1.2.6 Facteurs influençant la décision du déchargement de Calculs

Le déchargement réduit les limites des téléphones mobiles, mais il devient parfois très difficile de décharger une application ou un code vers le cloud en raison de certains facteurs qui influent sur la décision de déchargement sont :

✓ Taux de latence :

Il s'agit du délai entre le déchargement de l'application sur le cloud et le résultat qui revient sur le téléphone mobile. Ce taux de latence dépend essentiellement de la distance entre le téléphone mobile et le cloud. Si le mobile est très éloigné du cloud alors l'application mettra beaucoup de temps à se décharger et à renvoyer le résultat.

✓ Bande passante :

La bande passante désigne la vitesse de transfert des données sur le réseau. Le déchargement de toute application sur le cloud nécessite une vitesse de transfert de données élevée. La bande passante dépend de la liaison sans fil entre l'appareil mobile et le cloud. Différents utilisateurs utilisent différents types de réseau, ce qui affecte les besoins en bande passante du déchargement. Lorsque la connexion sans fil est excellente, une grande quantité d'exécution d'applications et de données doit être déchargée vers le cloud, mais lorsqu'elle est mauvaise, seule une petite quantité peut être déchargée pendant un temps limité.(Management Research Offloading Approach In Mobile Cloud Computing)

Si la connectivité réseau est mauvaise, l'application doit s'exécuter localement et si la vitesse du réseau est stable et élevée, l'application doit s'exécuter sur le serveur cloud car le déchargement nécessite une bande passante élevée du réseau.

✓ Hétérogénéité :

L'hétérogénéité désigne les différents types de réseaux sans fil utilisés par les appareils mobiles et les différents types d'appareils mobiles utilisant le déchargement. Parfois, le réseau devient si pauvre qu'il devient très difficile de décharger une application sur le cloud et de récupérer le résultat. Le cloud computing mobile possède divers réseaux, infrastructures, technologies d'appareils mobiles, architectures cloud et réseaux de communication, ce qui augmente l'hétérogénéité dans l'environnement de cloud computing mobile.(Management Research Offloading Approach In Mobile Cloud Computing)

✓ Taille de l'application :

La taille de l'application affecte également la décision de déchargement. Si nous devons décharger une grande application sur le cloud, elle a besoin d'une bande passante réseau élevée, d'un taux de latence réduit, d'une meilleure connectivité réseau et d'une capacité de traitement élevée du serveur cloud. Il est également très difficile de partitionner une grande application et de décider quelle partie de l'application doit s'exécuter localement sur l'appareil mobile et quelle partie à distance sur le cloud.

1.2.7 Déchargement dans le Cloud Computing et dans le Mobile CloudComputing :

Le mot « informatique mobile » désigne essentiellement l'informatique liée aux appareils mobiles. Pour dire plus clairement, l'informatique mobile traite des données et des calculs liés à des appareils qui ne sont pas fixes et peut modifier le réseau au fur et à mesure des besoins. Au contraire, le cloud computing est le calcul de tout un ensemble d'appareils mobiles et, en même temps, d'appareils immobiles. Ainsi, on peut dire que le cloud computing est un super ensemble de cloud computing mobile.

En utilisant la même analogie, on peut également en déduire que le déchargement dans le cloud computing n'est rien d'autre qu'un super ensemble du déchargement dans le cloud computing mobile. Toutes les architectures et algorithmes pour les deux cas restent les mêmes ; la seule différence qui entre en ligne de compte est la mobilité des appareils informatiques envisagés.

Cloudcomputingmobile, le réseau auquel ils accèdent peut également changer avec leur mouvement. Ainsi, dans le cas du déchargement dans un paradigme de cloud computing mobile, le changement de réseau est un facteur essentiel, comme avec le changement de réseau, le serveur auquel l'appareil accède change et la nécessité d'un calcul constant des techniques d'optimisation énergétique également surgit.

1.2.8 Sélection du chemin du cloud pour le déchargement

Le trafic de données sur les réseaux mobiles a connu une augmentation exponentielle au cours des dernières années, principalement en raison de la très grande popularité des tablettes, des smartphones et des ordinateurs portables. Comme le trafic de données sur les réseaux mobiles a considérablement augmenté, il est clair qu'il existe un besoin immédiat de décharger le trafic de données pour obtenir les meilleures performances des services voix et données. En conséquence, différentes méthodes, y compris des technologies telles que le Wi-

Fi, la mobilité de flux IP et les femtocells, ont été proposées pour gérer le trafic de données. Parallèlement au développement du cloud computing, le déchargement est devenu un moyen plus attrayant de prolonger la durée de vie de la batterie et de réduire le temps d'exécution sur les appareils mobiles. Mais un grand nombre de nuages apparaissent dans le ciel avec des charges et des conditions différentes, et le déchargement du même programme sur différents nuages peut entraîner différentes quantités de calcul dans la même durée en raison de différentes vitesses de nuage, et peut coûter un temps de communication différent en raison de la bande passante et disponibilité du nuage. Par conséquent, une méthode de sélection de chemin de nuage optimale est nécessaire lors du choix du meilleur nuage.

1.2.9 Méthodes de sélection de chemin de nuage

Il existe différentes manières de déterminer la paire de chemins cloud optimale lorsqu'il existe plusieurs chemins de ce type. Ils pouvez être résumé comme suit :

- Aléatoire : Sélectionnez la paire de chemins de nuage de manière aléatoire.
  - **Dépend de la bande passante :** choisissez la paire de chemins cloud avec la bande passante la plus élevée.
  - **Dépend du taux d'échec du lien** : sélectionnez la paire de chemins cloud avec le taux d'échec du lien le plus faible.
  - Dépend du facteur d'accélération : sélectionnez la paire de chemins de nuage avec le facteur d'accélération le plus élevé.
  - **En fonction du coût :** sélectionnez la paire de chemins cloud avec le coût le plus bas.

- Problèmes de sélection de chemin de cloud :

  La méthode de sélection du chemin du cloud doit être choisie en tenant compte des problèmes sous-estimés :

  - ✓ Bande passante : Elle dépend de la liaison sans fil entre les appareils mobiles et le cloud. Lorsque la connexion sans fil est excellente, une grande quantité d'exécutions d'applications et de données peut être déchargée vers le cloud, mais lorsqu'elle est mauvaise, seule une petite quantité d'exécutions d'applications et de données peut être déchargée pendant un temps limité.
  - ✓ Prix : Il dénote le coût pour la même quantité de calcul et sa variation dans les différents services cloud. Pour les fournisseurs de services de cloud mobile, comment créer un Le schéma de provisionnement des services est essentiel, en particulier lorsque la ressource cloud mobile est restreinte.
  - ✓ Vitesse : Il présente la vitesse à laquelle un serveur peut calculer sur le cloud. Relativement, il peut être mesuré par le facteur d'accélération F , qui compare la vitesse d'exécution du cloud à celle d'un appareil mobile.

- ✓ Sécurité : Tout d'abord, déplacer toutes les données et ressources informatiques vers le cloud est dangereux ; par exemple, le suivi des individus grâce à des données de navigation basées sur la localisation déchargée dans le cloud. En outre, les paramètres de sécurité et de confidentialité dépendent des fournisseurs de cloud, car les données sont stockées et gérées dans le cloud.
- ✓ Disponibilité **:**Elle est liée à la défaillance du lien et à l'indisponibilité du cloud pendant tout le processus de déchargement. Des défaillances peuvent survenir en raison de la mobilité des appareils mobiles et de la connectivité instable des liaisons sans fil, qui rendent les performances moins prévisibles d'un programme exécuté sous le contrôle de systèmes de déchargement.

1.2.10 La déchargement et la prise de décision :

Les avantages du déchargement dépendent principalement du moment où la décision de déchargement est prise, qu'elle soit effectuée au bon moment et de la bonne manière. Les décisions de déchargement peuvent impliquer plusieurs facteurs tels que la disponibilité des ressources et des composants, l'intermittence de la connectivité et la capacité du réseau. Il est très difficile de prendre une décision pour le déchargement, par exemple si le déchargement sera bénéfique ou non, quelle partie de l'application doit être déchargée. Déchargement les décisions pouvez être fait dans différents façons :

- ✓ Que décharger :

Que signifie le nom des tâches candidates à décharger ? Nous devons décider du nom de la tâche ou de l'application à décharger via le schéma de partitionnement.

- ✓ Où décharger :

Où signifie sélectionner l'espace cible où l'application va décharger. Il décrit le type de substitut et le choix de la cible de déchargement appropriée (par exemple local, cloudlet et □□cloud) dans lequel l'application doit être déchargée.

- ✓ Comment décharger :

Comment signifie sélectionner la stratégie appropriée qui peut être utilisée pour décharger l'applicationsur le nuage. Comment introduit des plans de déchargement qui permettent à l'appareil de planifier le déchargement

- ✓ -Quand décharger :

Quand signifie décider du moment du déchargement. Lorsque l'on considère les conditions de déchargement et les changements dynamiques de contexte, car parfois le déchargement peut ne pas valoir la peine du tout.

1.1 Conclusion

Le déchargement informatique est nécessaire pour décharger les tâches lourdes sur le cloud afin d'économiser l'énergie des appareils mobiles. De nombreuses applications doivent être exécutées simultanément sur l'appareil mobile. Parfois, certaines applications nécessitent plus d'énergie et de puissance de traitement, ce qui réduit la puissance et la vitesse de l'appareil mobile. Il est donc recommandé de décharger ce type de tâches en dehors de l'appareil mobile. , ces tâches s'exécutent là-bas, puis le résultat est renvoyé à l'appareil, grâce à cette technique, beaucoup d'énergie de l'appareil mobile peut être économisée.

Ainsi, nous pouvons dire que le déchargement est une technique qui surmonte certaines des limitations des téléphones mobiles. Certains des avantages de l'utilisation déchargement sont:

- ✓ Améliorer les performances :

L'utilisation du déchargement sur les appareils mobiles améliore les performances globales de celui-ci en exécutant ou en enregistrant certaines tâches lourdes en dehors de l'appareil mobile.

- ✓ Économisez de l'énergie :

Ledéchargement peut prolonger la durée de vie de la batterie de l'appareil mobile en migrant les travaux énergivores vers le cloud.

2 Chapitre 2 : Partitionnement et Optimisation d'énergie

2.1 Introduction :

MCC permet l'utilisation et l'accès aux services Cloud à l'aide d'appareils mobiles. Les appareils intelligents sont conçus pour être plus avancés et plus performants. Les développeurs créent des applications de plus en plus complexes pour répondre aux besoins des utilisateurs. Ces applications nécessitent des quantités croissantes de puissance et d'énergie de calcul. L'un des défis majeurs de l'évolution des appareils mobiles est l'équilibre entre une application gourmande en ressources et un appareil mobile aux ressources limitées. La durée de vie limitée de la batterie est un problème très important pour les appareils intelligents. Cependant, les appareils mobiles sont limités par leur capacité de stockage, leur puissance de traitement et leur autonomie limitées.

L'un des mécanismes les plus importants du MCC est le déchargement des tâches, où il permet la migration d'une tâche compliquée d'applications mobiles depuis des appareils mobiles vers un environnement cloud. Cependant, certains problèmes liés au processus de déchargement doivent être résolus, tels que la consommation d'énergie, la gestion du temps et la tolérance aux pannes.(Khadijah S, AnasBasalamah, Lo, & Mohammad)

2.1 Energy awarness :

En raison de la durée de vie limitée de la batterie d'un appareil mobile, la sensibilisation à l'énergie est devenue l'un des problèmes vitaux du MCC. Les appareils mobiles ont une source d'alimentation limitée et l'exécution de nombreuses applications entraîne constamment une grande décharge de la batterie. Ainsi, le profilage énergétique et l'estimation de la consommation d'énergie de l'appareil sont très importants. En MCC, la consommation d'énergie doit être inférieure au bénéfice obtenu. Les composants logiciels inutiles doivent

être déchargés et les composants énergivores doivent être redéployés vers l'hôte le plus ingénieux.

L'estimation de la consommation d'énergie est juste à l'opposé du profilage énergétique. C'est l'estimation préalable d'une application non pas pendant l'exécution mais au moment de la construction. Cette estimation est effectuée sur la base de certains paramètres de contexte tels que la taille des données échangées sur le réseau, les entrées aux interfaces des composants, la fréquence d'invocation des interfaces des composants, etc. L'équation pour l'estimation du coût de l'énergie est la suivante :

Coût énergétique du système = Coût énergétique de calcul + Énergie de communication + Coût énergétique global de l'infrastructure

/ Où (coût énergétique computationnel + énergie communicationnelle) fait référence au coût énergétique global.

2.1.1 Économie d'énergie

L'énergie est une contrainte majeure pour les systèmes mobiles [6]. Récemment, le développement de systèmes de smartphones dotés d'un grand nombre de fonctionnalités consomme plus d'énergie et raccourcit la durée de vie de la batterie. Le déchargement peut prolonger la durée de vie de la batterie en transférant les parties énergivores du calcul vers les serveurs.

La puissance consommée pour effectuer la tâche au sein du système mobile est donnée par :

$$E = p_m * \frac{w}{s_m}$$

La puissance totale consommée compte tenu de la transmission et du calcul est déterminée par :

$$E' = p_c\left(\frac{d_i}{B}\right) + p_i\left(\frac{w}{s_s}\right)$$ Où

(Di/B) est le temps de communication

(W/ss) est le temps de calcul côté serveur

Le déchargement permet d'économiser de l'énergie si la condition suivante est remplie :

$$E > E' \Rightarrow p_m * \left(\frac{w}{s_m}\right) > p_c\left(\frac{d_i}{B}\right) + p_i\left(\frac{w}{s_s}\right) \Rightarrow w\left(\left(\frac{p_m}{s_m}\right) - \left(\frac{p_i}{s_s}\right)\right) > p_c\left(\frac{d_i}{B}\right)$$

On observe que dans la zone où le calcul est élevé, un déchargement complet doit être utilisé et où un calcul modéré est requis, un déchargement partiel peut être effectué.(Mobile Cloud Computing Architectures Algorithms And Applications)

2.1.2 Taxonomie des approches de partitionnement des applications:

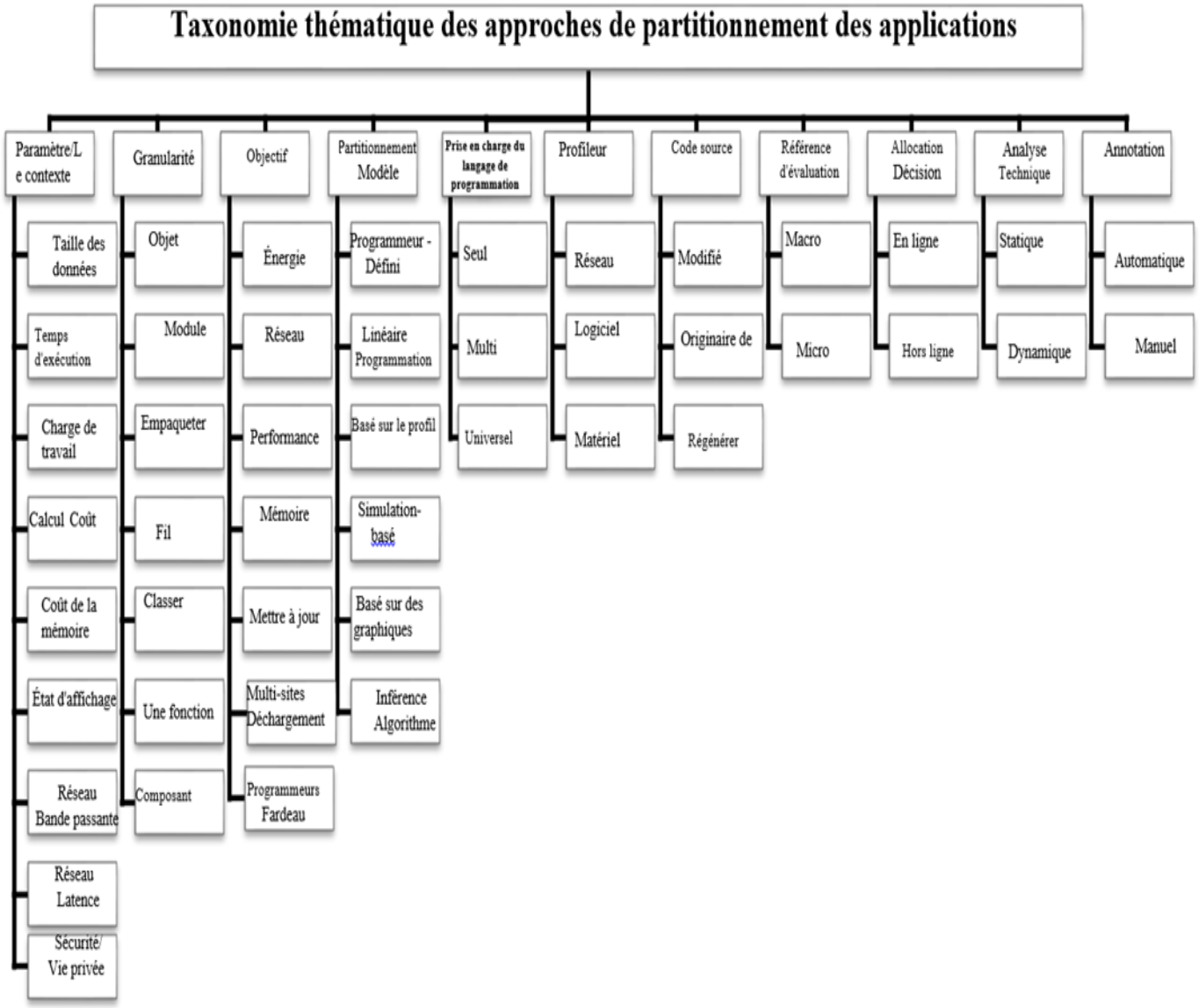


Figure 8 Thematic Taxonomy of Application

Attribut	Type

Granularité : indique le niveau de granularité pour partitionner une application mobile de calcul intensif.	• Module : L'ensemble de modules de l'application est cloisonné et distribué. • Méthode : Cloisonnement se produit au niveau des méthodes de l'application. • Objet : L'objet d'une application est partitionné pour se préparer à la recherche de nourriture cyber. • Thread : Cloisonnement se produit au niveau des threads d'une application. • Classe : L'application est divisée en classes pour la déportation. • Tâche : Application est partitionnée selon les tâches. • Composant : Partitionner un groupe de classes qui peuvent ou non être couplée. • Bundle : Groupes de classe d'applications Java. • Allocation-site : Cloisonnement se produit au niveau du lieu d'affectation où tous les objets de ce site particulier seront être considérés comme une seule unité. • Hybride : Les résultats de partitionnement sont constitués de granularité différente.
Objectif : est représenté par la fonction objective pour l'application de partitionnement et traitement d'applications distribuées	• Amélioration de la performance : les algorithmes de partitionnement ont tendance à augmenter leurs efficacités pour la mesure particulière (S). Ces mesures d'amélioration de la performance incluent le débit, le temps d'adaptation, et l'efficacité de l'algorithme et le coût la latence, le temps d'exécution, ainsi que la charge CPU. • Multi-sites de déchargement : L'application est divisée et distribuée entre plusieurs serveurs distants. • La réduction des contraintes de mémoire : ils ont tendance à atténuer le problème de la restriction de la mémoire sur le mobile. • Réduction de la charge du réseau : ils encourent une base surcharge du réseau pour le calcul de déchargement. • Réduire le fardeau des programmeurs : réduire l'effort de programmeurs dans le développement et le partitionnement d'applications. • Economies d'énergie : Un facteur crucial dans les algorithmes de partitionnement, réduire la consommation d'énergie tout en prolongeant la durée de vie de la batterie. • Mise à jour dynamique de l'application : L'application est partitionnée et mise à jour sans l'arrêter ou arrêter les serveurs distants.
Modèle : montre le type de partitionnement utilisé pour modéliser les composants d'applications	• modèle de partitionnement basé Graphe : APAs résument le processus de partitionnement entier comme un graphe. • modèle de Programmation linéaire (LP) : APAs

mobiles ou leurs capacités.	formulent des équations linéaires pour représenter le partitionnement de l'application. • modèle hybride : APAs combinent le modèle à base de graphes et le modèle LP. Néanmoins, un certain nombre d'algorithmes de partitionnement ne modélisent pas l'application comme équation LP ou graphe, par conséquent, de tels algorithmes sont classés comme APAs exceptionnelle.
Langage de programmation : indique le type de langage de programmation prise en charge par l'approche de partitionnement.	• La programmation unique : L'approche de partitionnement est limitée à un environnement de programmation. • Plusieurs langages de programmation : Une approche de partitionnement qui prend en charge plusieurs langages de programmation. • Support universel de langage de programmation : L'approche de partitionnement prend en charge la majorité des langages de programmation.
Profileur : indique le type de profilage utilisé par l'APA	• profileur matériel : Il recueille les informations pertinentes pour le matériel physique du dispositif mobile telles que le processeur, la mémoire vive, et la batterie. • profileur logiciel : Il recueille les informations de l'application tels que la taille des données accessibles (envoyer la taille, recevoir la taille, et la taille de transfert), interdépendance modulaire entre structures, le comportement de l'application, le coût de la performance (temps d'exécution, le débit et la latence) et la taille du code. • profileur réseau : Il rassemble des informations sur l'environnement réseau, par exemple, la connectivité Wi-Fi / 3G / 4G / Wi-Max, la bande passante, et le taux de transfert de données.
Décision d'attribution : indique la prise de décision ou la politique de l'APA dans l'allocation des composants (local/ à distance).	• en ligne, où la décision est prise lors de l'exécution. • hors-ligne, où la décision est faite avant l'exécution. • hybride, où une partie de la décision est faite par une spécification définie par le programmeur ou un outil d'analyse statique, et, une partie de la décision est faite lors de l'exécution de l'application.
Technique d'analyse : C'est la technique utilisée pour l'identification des relations de dépendances entre les composants d'une application mobile	• statique : elle est mise en oeuvre au niveau du source ou au niveau du byte code. • dynamique : elle implique le profilage d'exécution, et l'étude des composants l'interaction de l'application en cours d'exécution.
Annotation : c'est des métadonnées syntaxiques ajoutées au code source de l'application.	• automatique : le cadre de l'exécution met en oeuvre automatiquement l'annotation en utilisant le profileur pour recueillir l'information et annoter le composant pertinent concerné. • Manuel : annotation faite par les développeurs d'applications à la phase de conception, et elle consiste

	à examiner l'intensité et l'étendue des composants de l'application au moment de la conception.

Tableau 1Types des attributs APAs

2.1.3 Modèle basé sur des graphes :

Michel et al.Propose la première nouvelle approche pour produire des applications partitionnées, en modélisant l'application sous forme de graphe de dépendances, puis en partitionnant automatiquement les applications existantes sur la base d'une entrée à faible effort des développeurs et d'une analyse de code statique.(M, M , B , & M, 2012)

L'obtention de la décision de partitionnement optimale dans les approches basées sur les graphes est un problème NP-Complet.(Computers and Intractability : A Guide to the Theory of NP-Completeness)

La plupart des algorithmes sélectionnés se basent sur le modèle de graphe, nous avons différents types de graphes, les APAs actuels mettent en oeuvre les modèles de graphes suivants pour l'application de partitionnement :

- ✓ *Un graphe de granularité hybride (HGG)* pour représenter le temps d'exécution d'une application : est proposé par Abebe et al.pour réduire la surcharge du réseau. L'aspect critique est que cette approche peut fournir une adaptation de niveau plus fine et des topologies d'objets plus efficaces sans encourir les frais généraux de calcul d'un graphe au niveau de l'objet, HGG offre une granularité plus fine qu'un graphe de classe, ce qui se traduit par plus de flexibilité. De plus, HGG reste plus petit que le graphe d'objets et par conséquent plus faisable en termes de calcul.

- ✓ Un *graphe de classe abstrait*, est proposée par Abebe et al.. Une autre nouvelle représentation de graphe distribué. L'aspect critique est que cette approche permet au résultat de la partition sur le cloud d'être rechargé sur le périphérique client à des fins utilitaires. De plus, l'efficacité des partitions générées est également améliorée car les coûts de couplage et de migration d'objets distants sont réduits. Le problème ici est qu'il n'a pas encore été testé dans des applications réelles.(E. Abebe et C. Ryan, 12-2012)

- ✓ L'idée *d'objets cassables (BoB)* pour le partitionnement d'application au moment de la compilation en Java est introduite par Jamwal et al. Les BoB sont les entités d'une application qui peuvent être facilement divisées. L'aspect critique de cette approche est qu'elle atténue les défis de l'interopérabilité et de la sensibilité au contexte. Cependant, il oblige les programmeurs à modifier le code source des applications qui n'est pas toujours disponible. En outre, les programmeurs ont besoin de connaissances sur les applications pour déterminer où il est nécessaire d'effectuer les modifications pour le partitionnement. (Iyer, 2005. APSEC'05 12e Asie-Pacifique , 2005).

- ✓ *Le graphe de flux de données :* représente les dépendances de données entre un certain nombre d'opérations.

- ✓ Le graphe de contrôle de flux de tâches : contrôle du flux du programme dont les noeuds représentent les tâches.

- ✓ Le graphe de contrôle de flux : indique tous les chemins qui pourraient être traversés par une application mobile lors de son exécution.

- ✓ Le graphe d'appels : un graphe orienté qui représente les relations entre les méthodes (les appels) dans une application mobile.

- ✓ Le graphe de dépendance : un graphe orienté représentant les dépendances entre plusieurs noeuds.

2.1.4 La programmation linéaire (LP)

Peut modéliser et formuler l'application comme un problème d'optimisation mathématique où certaines ou toutes les variables sont limitées à des nombres entiers. Dans la plupart des cas, LP sera utilisé pour optimiser les problèmes formulés qui représentent l'application mobile. Semblable au partitionnement d'application de modèle basé sur des graphes, la première étape consiste à vérifier si une annotation est nécessaire. Si oui, il faudra peut-être que le programmeur spécifie l'annotation manuellement. Si non, il continuera à vérifier si la fonction de profileur est implémentée dans l'application. Si oui, le profiler rassemblera les informations pertinentes qui ont demandé l'application. Si non, il passera à l'étape suivante pour formuler le problème LP. L'annotation par le programmeur et le résultat du profilage peuvent être utiles pour aider à la formulation du problème LP. Après cela, des

techniques de programmation linéaire, à savoir la programmation linéaire entière (ILP), la programmation linéaire zéro-un (0-1LP) et la programmation linéaire mixte entière (MILP), sont mises en œuvre pour résoudre le problème d'optimisation formulé.
Enfin, le résultat de l'optimisation est fourni à l'algorithme d'inférence pour décider et effectuer le partitionnement. La figure 1 montre l'organigramme de niveau abstrait du modèle basé sur LP partitionnement des applications.

2.1.5 Modèle hybride :

MAUI :

Est un système qui fournit un déchargement de code à grain fin pour optimiser les économies d'énergie avec une charge minimale pour le programmeur.(E. Cuervo, 2010)

MAUI fournit un environnement de programmation permettant aux programmeurs de produire un partitionnement initial de l'application via l'annotation des méthodes d'application qui sont déchargées pour une exécution à distance. Les programmeurs ont besoin de méthodes d'annotation et classes comme étant distantes, ce qui indique que le runtime MAUI doit envisager de décharger sur un serveur distant. Par conséquent, les programmeurs n'ont pas besoin de deviner s'il est logique ou non de décharger une méthode particulière. L'annotation en tant que local peut entraîner moins d'annotations, mais elle favorise les performances par rapport à l'exactitude de l'application.

Au lieu de cela, les erreurs sont nulles car la négligence de l'étiquetage dans l'approche MAUI n'affecte que les performances du programme et non son exactitude. MAUI promet une grande fiabilité.

CloneCloud :

Essaie d'analyser un processus d'application donné à qui exécution sur le cloud. L'aspect critique est qu'il prouve l'efficacité de l'analyse statique du code Java pour partitionner dynamiquement les applications. Cependant, CloneCloud ne prend en compte que les conditions d'entrée/environnementales limitées dans le prétraitement hors ligne. De plus, il doit fonctionner l'analyse à nouveau pour chaque nouvelle application créée. Étant donné que le concept de réutilisabilité des fonctions

et de couplage lâche n'est pas pris en compte, la composition logicielle n'est pas prise en compte adressée. En outre, il n'est pas facile pour les utilisateurs profanes de personnaliser les fonctionnalités de l'application car l'application doit être modifiée par programme.

2.2 Partitionnement d'application :

2.2.1 Définition

Le partitionnement d'applications est une technique de fractionnement de l'application en composants séparés, tout en préservant la sémantique de l'application originale .la déportation de calculs utilise le partitionnement de l'application mobile pour séparer la logique du fonctionnement de l'application mobile en partitions distinctes, qui sont capables de fonctionner de façon autonome dans un environnement distribué. L'application de la source d'origine peut ou ne peut être conçu, mis en oeuvre et déployé pour fonctionner sur un système standalone. Les composants résultants, cependant, sont distribués pour tirer parti de la plate-forme d'exécution distribuée. La figure 2.2 illustre le flux général des opérations impliquant le partitionnement des applications et la déportation de composants pour le MCC. L'application s'exécute sur le dispositif mobile et le mécanisme de profilage d'application évalue l'utilisation des ressources informatiques, la disponibilité des ressources et des besoins informatiques de l'application mobile. Dans une situation critique quand il y a une insuffisance de ressources sur le mobile, le mécanisme de résolution est activé pour séparer les composants à calcul intensif de l'application lors de l'exécution. Le mobile négocie avec les serveurs de Cloud pour la sélection d'un noeud de serveur approprié et la partition intensive de l'application est confiée au noeud distant pour le traitement à distance. Lors d'une exécution réussie des composants à distance de l'application, le résultat est renvoyé à l'application principale en cours d'exécution sur le dispositif mobile(: O. R. D. J. I. K. a. G. A. Ioana Giurgiu, 2009)

2.2.2 Comparaison des approches de partitionnement :

Approches	PG	GM	PLS	AD	AT	AN
Application de flux de données [47]	composant	flux de données	plusieurs	en ligne	dynamique	automatique
Graphique de granularité hybride [29]	hybride	granularité hybride	Seul	en ligne	statique	automatique
Graphe de classe abstrait distribué [30]	classe	classe	Seul	hors ligne	dynamique	N / A
OLIE [35]	classe	classe	Seul	en ligne	dynamique	Manuel
Partitionnement adaptatif multi-contraintes [5]	classe	multi-coût	Seul	en ligne	dynamique	N / A
Refactoring automatisé [36]	hybride	Dépendance interne	Seul	hors ligne	dynamique	Manuel
Déploiement dynamique du logiciel [34]	bundle	consommation	Seul	hybride	dynamique	automatique
J-Orchestre [43]	objet	N / A	Seul	hors ligne	dynamique	Manuel
Analyse paramétrique [45]	tâche	flux de contrôle des tâche	universel	en ligne	dynamique	Manuel
RCMCPP [46]	module	N / A	N / A	en ligne	dynamique	automatique
Déchargement de calculs multi-sites [41]	site d'allocation	interaction avec un objet	Seul	en ligne	statique	automatique
Appeler le nuage [27]	bundle	flux de données	plusieurs	hybride	dynamique	automatique
Clone Cloud [33]	thread	flux de contrôle	plusieurs	en ligne	statique	automatique
MACs [37]	bundle	N / A	Seul	en ligne	dynamique	automatique
Amélioration de l'efficacité énergétique [40]	composant	N / A	universel	hors ligne	dynamique	Manuel
MAUI [26]	méthode	appel	Seul	en ligne	dynamique	Manuel
Application de partitionnement [42]	composant	dépendance	plusieurs	hors ligne	statique	Manuel
Mises à jour dynamiques [25]	composant	N / A	Seul	hors ligne	dynamique	Manuel
Roam [32]	composant	N / A	Seul	en ligne	statique	Manuel
Prédiction de complexité [39]	composant	N / A	Seul	en ligne	statique	Manuel
Simplifier le cyber-foraging [90]	module	tâche	universel	hors ligne	statique	Manuel
Partitionnement logiciel énergétiquement optimal [67]	tâche	tâche	N / A	hors ligne	dynamique	automatique
Wishbone [68]	thread	flux de données	plusieurs	hors ligne	statique	Manuel

Tableau 2 Comparaison générale des approches de partitionnement des applications

2.2.3 RelatedWork :

En MCC, plusieurs études ont récemment été discutées dans la littérature, nous en présentons quelques-unes, avec les problèmes les plus importants que les chercheurs ont voulu résoudre.

- Lee et alprésentent un algorithme de planification de la tolérance aux pannes basé sur un groupe dans Mobile Grid. Cet algorithme classe les appareils mobiles en groupes d'appareils mobiles en fonction des disponibilités complètes et partielles et utilise un algorithme de réplication adaptatif pour la tolérance aux pannes. Les résultats expérimentaux montrent que leur algorithme d'ordonnancement offre une performance supérieure à celui qui n'utilise qu'une prédiction de disponibilité. Tout au long des expériences, ils ont constaté que la tolérance aux pannes active (c'est-à -dire la réplication) est essentielle pour améliorer les performances du réseau mobile. À l'avenir , ils prévoient de mener une plus grande variété d'expériences pour étudier des facteurs supplémentaires qui contribuent aux performances du réseau mobile.(Lee)

- Raju et alDans cet article, une architecture DRFT est proposée pour gérer les machines virtuelles défaillantes dans l'environnement MCC. Le mécanisme est inspiré du système de résistance aux maladies humaines appelé DR -Approach. Basée sur le système de résistance aux maladies humaines, l'approche a été développée avec quatre modules importants tels que le module de surveillance, le module de réponse, le module de connaissances et le module de mémoire. Ces quatre modules collaborent ensemble pour résoudre le problème de réordonnancement des ressources dans l'environnement MCC. Cet article considère trois algorithmes bien connus pour l'ordonnancement tels que HEFT, Min-Min et GA pour la série de simulations et la performance de l'approche DRFT est comparée à l'approche de tolérance aux pannes basée sur le groupe dynamique (DGFT). Les résultats ont prouvé que le DRFT avait des performances supérieures pour gérer les machines virtuelles défaillantes dans différentes conditions.(by Dasari Naga Raju)

- Abd et alCet article a proposé un mécanisme d'isolement génétique amélioré basé sur le niveau de priorité et les combinaisons d'ADN pour réduire l'énergie consommée, gérer le temps de traitement et éviter l'échec de la tâche. Les travaux futurs comprennent la mise en œuvre du système et la publication des résultats du système suggéré. En outre, améliorez la technique proposée pour obtenir d'autres mesures de performance telles que le test du trafic réseau et la sécurité des tâches déchargées.(by Sura Khalil Abd)

- Al-Sayed et alDans cet article, Deux mécanismes de surveillance développés pour prédire les états surveillés des ressources cloud. Ces mécanismes sont des mécanismes basés sur CTMC et DTMC. Les deux mécanismes envoient des mises à jour au consommateur lorsque la différence entre les valeurs prévues et réelles dépasse la valeur ETD. Ces mises à jour sont utilisées pour régler le modèle de prédiction au niveau du consommateur. Dans les deux mécanismes, un modèle de Markov à l'état K est développé sur la base des données d'apprentissage.(by Mustafa M. Al-Sayed)

- Peng et al. Cette étude combine la technique DVFS pour réduire la consommation énergétique des appareils mobiles sous les contraintes de temps de service des utilisateurs. Cette étude a formulé le temps de réalisation et la consommation d'énergie, et leur donne les poids selon l'optimisation conjointe. Grâce à de multiples expériences, les équations contenant le temps de réalisation et la consommation d'énergie peuvent être utilisées comme fonction objectif de WOA . Les deux objectifs d'optimisation sont combinés à l'aide d'une méthode de sommation pondérée, et la solution est finalement atteinte dans une plage raisonnable, ce qui peut améliorer le bénéfice total de l'ensemble du système. Et les performances dans les mêmes conditions expérimentales sont respectivement comparées à deux autres algorithmes de planification de tâches (by Hua Peng a)

- Tang et la planification des tâches dans MCC est connue comme un problème NP-difficile, qui a attiré beaucoup d'attention au cours des dernières années. cet article modélise ce problème comme un problème d'optimisation de la consommation d'énergie, tout en tenant compte de la dépendance des tâches, de la transmission de données et de certaines conditions de contrainte telles que le délai de réponse et le coût, et le résout en outre par trois algorithmes heuristiques différents. Les résultats expérimentaux ont montré que notre stratégie de planification des tâches soucieuse de l'énergie peut réduire considérablement la consommation d'énergie sur l'appareil mobile.(by Chaogang Tang)

- Lin et al. Ce travail étudie le problème d'ordonnancement des tâches MCC. À notre connaissance, il s'agit du premier travail de planification de tâches qui minimise la consommation d'énergie sous une contrainte de temps d'exécution stricte pour le graphe de tâches dans l'environnement MCC, en tenant compte de la planification conjointe des tâches sur les cœurs locaux de l'appareil mobile, de la communication sans fil canaux et le cloud. Un nouvel algorithme est proposé qui part d'un ordonnancement à délai minimal et effectue ensuite une réduction d'énergie en migrant les tâches entre les cœurs locaux et le cloud. Un algorithme de réordonnancement en temps linéaire est proposé pour la migration des tâches de manière à réduire efficacement la complexité de calcul globale. Les résultats de la

simulation démontrent une réduction d'énergie significative avec la contrainte de temps d'achèvement globale satisfaite.(Xue Lin)

- Guo et al. ce travail est le premier travail d'intégration du déchargement dynamique avec la planification des ressources afin d'atteindre la minimisation de la consommation d'énergie conjointe et du temps d'exécution de l'application sous la contrainte de délai d'achèvement et l'exigence de priorité des tâches. un nouvel algorithme distribué eDors proposé qui est composé des sous- algorithmes de sélection de déchargement de calcul, de contrôle de fréquence d'horloge et d'allocation de puissance de transmission. l' algorithmeeDors implémenté dans un banc d'essai réel et les résultats expérimentaux démontrent que par rapport aux politiques de déchargement existantes, l' algorithme eDors peut réduire efficacement la consommation d'énergie et le temps d'exécution de l'application, en tirant parti du contrôle de la fréquence d'horloge du processeur dans le calcul local et de la puissance de transmission allocation dans le cloud computing.(by Songtao Guo).

✓ Comparaison de certaines solutions proposées liées au MCC dans la littérature :

N°	Articles proposés	Algorithme utilisé	Taux de défaut	Makespan Time	Optimisation d'Energy	Déchargement	Hétérogénéité	Control Messages	Stockage	% des taches executées
1	Lee et al.	Group based fault tolerance	□	□	-	□	□	□	-	-
2	Raju et al.	Disease Resistance Approach	□	□	-	-	□	-	-	□
3	Abd et al.	k-out-of-n framework (denoted byKNF)	□	□	□	-	□	-	-	□
4	Park et al.	MARKOV	□	□	□	□	-	□	□	□

	[24]	chain based monitoring Model								
5	Al-Sayed et al. [32]	Dynamic Grouping Technique	□	□	-	-	-	-	-	-
6	Kashanchi et al. [33]	A genetic method for task scheduling	□	-	-	□	-	-	□	-
7	Peng et al. [34]	Reliability-compliant and Energy-aware Data Storage	□	-	□	-	□	-	□	-
8	Tang et al. [35]	Energy-Efficient Task Scheduling	□	-	□	-	-	-	-	□
9	Lin et al. [36]	Performance -Aware Task Scheduling	-	-	□	□	-	-	-	□
10	Guo et al. [37]	EETS. Model for Task Scheduling	-	-	□	□	-	□	-	-
11	Wei et al. [38]	MLMCM for Task Scheduling	-	□	-	-	□	-	-	-
12	Nawrocki et al. [39]	M L through Adoptive service	-	□	-	-	-	□	□	-

13	Akki et al. [40]	N.N. based optimization methods	-	-	□	□	-	-	-	□
14	Shakarami et. al. [41]	stochastic-based offloading approaches	-	□	□	□	-	□	-	□

Tableau 3 Comparaison de certaines solutions proposées liées au MCC dans la littérature

2.1 Conclusion :

L'exécution d'applications mobiles à l'aide de ressources cloud de calcul est la dernière stratégie d'augmentation des contraintes de ressources des appareils mobiles intelligents (SMD : Smart Mobile Devises). Le déchargement informatique nécessite que l'application mobile soit partitionnée lors de l'exécution de l'application sur les SMD. Puisqu'une approche de partitionnement optimale promet une optimisation des économies d'énergie et des performances sur les CMS, le partitionnement de l'application mobile au moment de l'exécution est une perspective de recherche difficile. Ce chapitre passe en revue les approches de partitionnement des applications (APA : Application PartitioningAproach) existantes dans différents domaines, y compris le Mobile Cloud Computing (MCC). Cette recherche est motivée par l'objectif de mettre en évidence les problèmes et les défis associés aux APA existants dans la gestion de différents contextes, l'utilisation des ressources locales et cloud pendant le partitionnement et l'augmentation de l'exécution des applications mobiles à forte intensité de calcul. Les points communs et les écarts dans ces approches sont analysés sur la base de paramètres significatifs tels que la sensibilité au contexte, le niveau de granularité, l'annotation et le modèle de partitionnement. Enfin, nous avons mis en avant les problèmes de recherche ouverts dans les approches de partitionnement d'applications pour MCC qui restent à étudier

Le mobile cloud computing est basé sur l'utilisation des dispositifs mobiles,Qui possèdent des limitations au niveau de l'énergie des batteries. Pour augmenter la durée de vie de ces dernières et améliorer la performance des applications, nous avons vu dans le chapitre précédant que la déportation du code vers des serveurs Cloud sera la solution, il existe plusieurs approches proposées pour gérer ce mécanisme mais il reste beaucoup de paramètres à prendre en considération et des améliorations à faire.

3 Chapitre 3 : Approche Proposée

3.1 Introduction

Dans ce chapitre, nous présenterons quelques étapes ou le code source, donnant une vue d'ensemble de la structure de l'application. Dans un premier temps, nous déterminons les différents composants, ensuite nous présenterons l'algorithme proposé, en illustrant l'exécution par des captures d'écran.

3.1 Description Générale de l'approche :

L'approche proposée est basée sur les algorithmes hybrides qui combinent les avantages des deux types de modèles LP et graphe, plus spécifiquement notre approche utilise une source aléatoire pour produire une solution correcte avec une bonne probabilité de partitionnement et déportation sur le Cloud. Le but principal est de réduire le temps d'exécution des différentes taches de l'application mobile, et par conséquent optimiser l'énergie du dispositif Mobile.

L'approche tente à atteindre la solution optimale et dynamique en considéré les couts locaux dans le mobile, les couts à distance dans le Cloud et les couts de communication entre le dispositif mobile et Cloud, par l'utilisation des différents modules profileur, analyseur et solveur qui vont déterminer le meilleur résultat de partitionnement.

L'approche proposée suit les étapes de déchargement de calculs cités dans chapitre précédent :

- ***Partitionnement de l'application*** : Diviser l'application en des méthodes en utilisant un algorithme et déterminer en même temps les méthodes appelées par chaque méthode et le niveau de pondérations de chaque méthode à base de ses dépendances

des autres méthodes de l'application. Il est nécessaire de déterminer les méthodes fixes qui ne doivent être jamais déportées.

- ***Préparation :*** Déterminer le temps nécessaire pour l'exécution local ou dans le Cloud de chaque méthode
- ***Décision de déchargement :*** L'étape finale qui sert à prendre une décision de déchargement

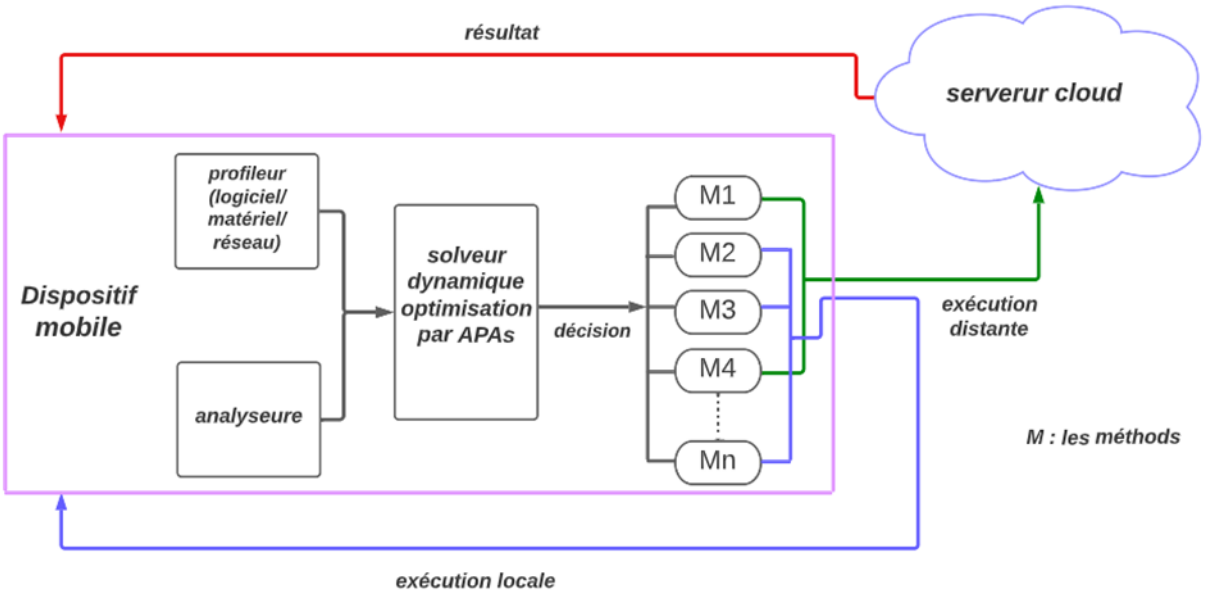


Figure 9 le Comportement général dans un processus de déportation

3.1.1 Objectifs de l'approche proposée

Comme la plupart des travaux réalisés dans le domaine de déchargement de calculs, notre approche vise à optimiser la consommation d'énergie en minimisant le temps d'exécution de l'application Mobile :

- **Minimiser le temps d'exécution** : le but principal c'est d'obtenir le temps d'exécution minimal des méthodes, pour cela il faut calculer le temps d'exécution local dans le dispositif mobile et à distance dans le Cloud, en prenant en considération le temps de communication entre le Cloud et le dispositif mobile. Dans ce scénario la décision de déchargement sélectionne alors le meilleur partitionnement qui minimise le temps d'exécution total de l'application mobile.
- **Optimiser l'énergie** : Ce principe est basé sur la fonctionnalité de profilage qui a évalué à la fois les ressources de Cloud et les ressources actuelles du mobile (CPU, RAM, batterie). Lorsque le dispositif mobile ne dispose pas de ressources suffisantes de l'application mobile, ce dernier est partitionné et les parties qui

nécessitent des calculs intensifs sont déchargées dynamiquement au moment de l'exécution vers le Cloud, après l'exécution réussie des composants distants de l'application le résultat est renvoyé à l'application principale sur le mobile. Ces calculs montrent que l'offloading est bénéfique pour l'optimisation d'énergie des systèmes mobiles lorsque l'application nécessite de grandes quantités de calcul avec des temps de communication relativement courts (secondes nécessaires pour transmettre et recevoir).

Suivant le tableau des APAs présentés dans le chapitre 2, nous avons :

APA	Granularité	Modèle	Langage supporté	Profile	Décision d'allocation	Technique d'analyse	annotation
Approche proposée	Méthode	Hybride	single	Réseau Logiciel Matériel	Online	dynamique	automatique

Tableau 4 Caractéristiques de l'approche.

- **Granularité**: un type de partitionnement par méthode qui est le plus approprié dans le cas d'un langage de programmation orienté objet ;
- **Langage supporté** : le langage java ;
- **Modèle** : hybride, une première représentation générale se fera par un graphe d'appels de méthodes et la formulation du problème sera en LP.
- **Profile** : permet de récupérer et collecter l'ensemble des données concernant l'état matériel, logiciel et réseau.
- **Décision d'allocation** : Online, faudra prendre en compte le contexte de l'application.
- **Technique d'analyse et annotation** : se font d'une manière automatique ce qui permettra d'analyser les dépendances entres les méthodes, sélectionner et annoter celles qui sont les plus appropriées et qui vérifient les conditions et contraintes de déportation vers des serveurs Cloud.

3.2 Description détaillée de l'approche

Nous proposons une architecture qui sert à minimiser le temps d'exécuion et par conséquent garantit l'optimisation de la consommation d'énergie. L'architecture repose sur Trois modules qui décident quelles sont les méthodes qui vont être resté sur le dispositif mobile et quelles sont les méthodes qui doivent être déportées.

3.2.1 Module profileur

Ce modulesurveille l'ensemble de matériel et logiciel, le profilage utilisé est définie comme suit :

- ✓ **Profilage Matériel** : Ce module a pour but de collecter toutes les informations sur les ressourcesmatérielles physiques (Dispositifs mobile et Cloud), telles que la mémoire vive, le processeur, et la capacité de batterie ;
- ✓ **Profilage Logiciel** : Ce module est responsable de la collection des informations sur l'application mobile, telle que la taille de code
- ✓ **Profileur réseau** : ce module recueille les informations sur l'environnement réseau, par exemple, la bande passante, 3G ,4G et 5G, la connectivité Wi-Fi.

3.2.2 Module analyseur

Ce module construit un graphe G = (V;E) qui divise l'application selon ses différentes méthodes où les nœuds sont lesméthodes V = (v_1; v_2;…; v_N), et les arcs E(v_i; vj) sont les relations entre ces méthodes , où les sommets v_i et v_j sont des noueds voisins. Il est à noter que les méthodes ont deux valeurs différentes : la première est le coût d'exécution local dans le dispositif mobile et la deuxième est le coût d'exécution à distance dans le Cloud , et chaque arête prendre le coût qui dénote lecoût de communication entre les méthodes en cas de déchargement aux serveurs Cloud.

Le fonctionnement de ce module est illustré dans l'algorithme suivant :

Algorithme 1 : Analyseur

Input: Code source de l'application **;**
Output: Matrice des méthodes, temps d'exécution local et à distance de chaque méthode et coût de communication ;
if le code est compilé**then**
Décompilerle code ;
End if ;
Lire le code ;
forcode**do**
Rechercher des méthodes ;
Mettre le resulat dans une matrice ;
end for
for Matrice **do**
Calculer le temps d'exécution local
Calculer le temsp d'exécution à distance
Calculer le cout de commmunication
Mettre le resulat dans la meme matrice ;

end for
returnMatrice

Le résultat de l'algorithme donne un graphe de pondération:

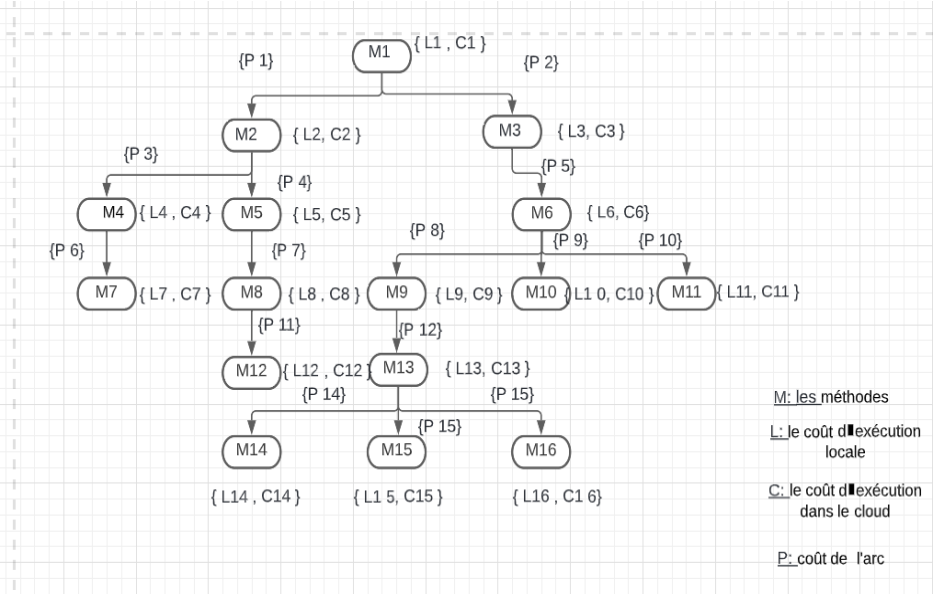


Figure 10 Graphe de pondération

3.2.3 Module solveur

Afin d'avoir un bon partitionnement qui minimise la consommation d'énergie, le module principal « Solveur » recueillie les informations des différents modules : profileur et analyseur, et les utilisent comme entrées pour une optimisation globale de la solution.

Avant d'expliquer l'algorithme et pour compléter notre approche, Nous présentons la formule utilisée par le solveur pour estimer l'énergie consommée :

$$E_{Gagnée} = \sum_{i=1}^{n} E_{Calcul\ local_i} - \sum_{i=1}^{n} (E_{Transfert_i} + E_{Calcul\ local_j}) > 0$$

n : méthodes en dépendances sinon c'est 1

Tel que :

Données transférées = (Envoyées + Reçues)

$$E_{Calcul\ local} = \frac{\text{Nombres d'instructions}}{\text{Instruction par seconde (fréquence de l'horloge)}}$$

$$E_{Calcul\ local} = E_{CPU\ en\ chômage} + \frac{\text{Données transférées}}{\text{Bande passante}}$$

L'énergie gagnée par ce processus sera l'énergie estimer localement moins l'énergie en cas d'exécution à distance d'un ensemble de méthodes plus l'énergie des méthodes restantes (qui s'exécuteront localement). L'estimation de l'énergie locale est égale aux nombre d'instructions

par méthode divisé par la vitesse du processeur (fréquence de l'horloge pour exécuter une seule instruction) le tous N fois pour l'ensemble des méthodes. L'énergie de transfert est égale à l'énergie de chômage du CPU au niveau du dispositif plus la division des données transférées (envoyées et reçues) par la capacité de la bande passante. L'énergie Cloud est égale aux nombre d'instruction divisé par la vitesse actuelle du serveur Cloud (est considérée négligeable).Enfin nous n'ajoutons pas le temps d'exécution de l'architecture parce qu'elle sera lancer quelque soit, même dans le cas d'une exécution purement locale.

Parmi la liste des méthodes envoyés par le module analyseur,le solveur décide quelle sont les méthodes qui doivent ètre exécutéessur le dispositif mobile et celles à distance, cette décision est soumise à certaines conditions, essentiellement le temps d'exécution, l'application de cette stratégie est difficile sans avoir un algorithme précis de partitionnement, alors nousallonsextraire les couts locales et à distance de chaque méthode et calculer le cout minimal de l'exécution de chacune en prenant en compte le cout de communication entre le dispositif mobile et le Cloud comme un facteur crucial.

Algorithme utilisé par ce module est montré dans la figure suivante :

Algorithme 2 :APA

Input:N Methodes, Coût d'exécution local de chaque méthode, Coût d'exécution à distance de chaque méthode, Coût de communication ;

Output: Coût minimum d'exécution, méthodes exécutées localement, méthodes exécutées à distance

Init ← Coût d'exécution local de tout les méthodes

Répéter

Générer un nombre aléatoire de méthodes **Nbre** (entre 1 et N) pour exéution locale

Généreraléatoirement **Nbre** méthodes

Alt ←Coût d'exécution local de **Nbre** méthodes + Coût d'exécution à distance des méthodes restantes (N-Nbre) + leurs Coûts de communication

if $Alt < Init$ **then**

$Opt \leftarrow Alt$

$Ini \leftarrow Opt$

end if

Jusqu'à K ;

return *Opt ;*

La première étape de l'algorithme sert à calculer une solution initiale qui est le coût de calcul de toutes les méthodes en cas d'exéution locale. Puis générer un nombre aléatoire de méthodes et les determiner ainsi aléatoirement. Dans cette étape, le coût d'exécution de la solution générée est calculé afin de le comparé avec le cout de la solution initiale. Si ce coût est plus petit elle sera determinée comme une solution optimale. L'algorithme sera répéter K fois (la valeur de cette variable est fixé lors de l'implémentation de l'algorithme)

L'obtention de la solution ou d'une solution acceptable répondant à des critères fixés est trouvée à la suite d'itération de cet algorithme.Après la prise de décision de ce module par le biais de cet algorithme, l'exécution de l'ensemble des méthodesest lancé.

❖ ***Scénario d'interaction entre les modules***

Voici le digramme de séquence qui représente les interactions entre les différents modules de l'approche proposé

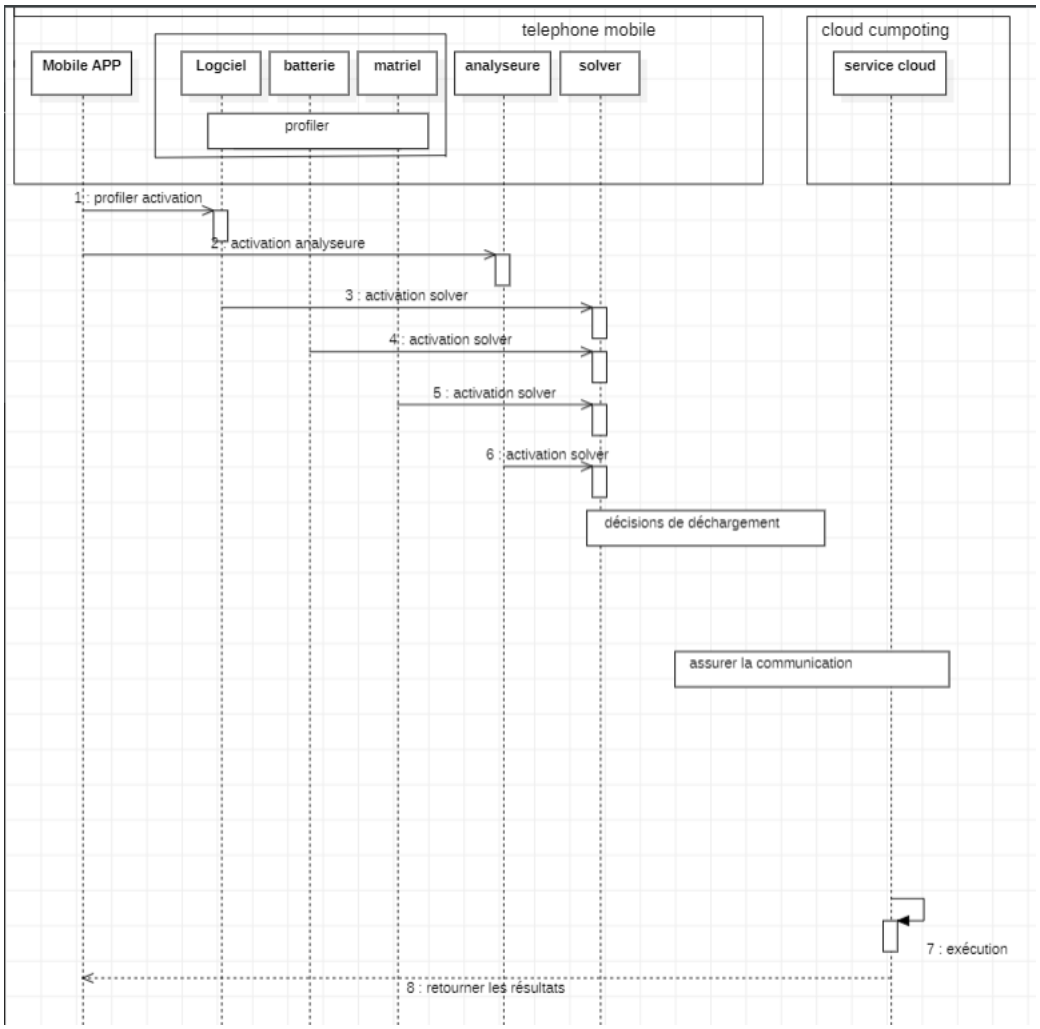


Figure 11:diagramme de séquence montrant l'interaction entre les différents modules

3.1 Conclusion

Dans ce chapitre nous avons présenté d'une manière générale de notre approche proposée puis le schéma détaillé de son architecture et l'acheminement des actions,

les différents modules qui la compose et leurs interactions par le biais du diagramme de séquence détaillé.

Le prochain chapitre sera consacrée à l'mplémentation et évaluation de l'approche proposée en détails.

4 Chapitre 4 : Implémentation et évaluation de l'approche proposée

1 Introduction

Dans le but d'optimisation de la consommation d'énergie et de temps d'exécution dans l'informatique mobile, nous avons proposé une approche qui se base sur le partitionnement des méthodes, les moins consommant d'énergie et de temps seront garder au niveau de dispositif mobile et déporter les autresméthodes une pour exécution distante au niveau de Cloud.

Dans ce chapitre nous avons appliqué notre algorithme sur deux applications mobiles sous Android Tic-Tac-Toe, elle appartient au domaine de jeux mobile, et une application de reconnaissance faciale.

Dans ce qui suit,nous allons présenter le domaine choisi, les outils et les langages utilisés pour le développement, et voyons les résultats de l'exécutionobtenus.

1.1 Présentation du domaine

Le jeu mobile est un excellent exemple d'adoption réussie du mobile et du nombre croissant de plates-formes dans l'industrie des médias et du divertissement, cependant, le jeu sur mobile rencontre aussi un certain nombre de défis comme l'utilisation excessive de ressource de dispositif mobile ce qui conduit à la mauvaise consommation de la batterie, la saturation de RAM et défaillances de CPU.

La reconnaissance des visages a été largement appliquée dans les applications du monde réel telles que la protection, la sécurité et la vérification de l'identité, grâce à le développement de dispositifs mobiles, nous pouvons bénéficier d'une méthode d'identification supplémentaire via un logiciel intelligent basé sur des techniques de vision.

1.2 Outils et langages utilisés pour l'implémentation de l'application

Afin d'implémenter notre approche nous somme basé sur le langage java, kotlin , l'IDE Android Studio et Nodejs

1.3 Présentation de l'IDE et Langage utilisés

Pour le développement de l'approche nous nous somme basé sur le langage java, kotlin et l'IDE Android Studio.

- JAVA : Java est un langage de programmation et une plate-forme de développement. Il réduit les coûts, raccourcit les délais de développement, stimule l'innovation et améliore les services applicatifs. Avec des millions de développeurs exécutant plus de 51 milliards de machines virtuelles Java dans le monde, Java continue d'être la plate-forme de développement de choix pour les entreprises et les développeurs.(ref : https://www.oracle.com/java/)
- Android studio : Android Studio est l'environnement de développement intégré officiel du système d'exploitation Android de Google, basé sur le logiciel IntelliJ IDEA de JetBrains et conçu spécifiquement pour le développement Android.(ref : https://developer.android.com/)

1.4 Présentation du Système d'exploitation mobile choisi

La puissance pour des écrans de toutes tailles

« Android est une plate-forme personnalisable et facile d'utilisation, présente sur plus d'un milliard d'appareils dans le monde entier : téléphones, tablettes, montres, téléviseurs, voitures, et ce n'est pas fini ! », sur le site officiel.

1.4.1 Android:

Android est un système d'exploitation mobile, basé sur le noyau Linux et développé actuellement par Google. Le système a d'abord été conçu pour les smartphones et tablettes tactiles, puis s'est diversifié dans les objets connectés et ordinateurs comme les télévisions (Android TV), les voitures (Android Auto), les ordinateurs (Android-x86) et les smartwatch (Android Wear). Le système a été lancé en juin 2007 à la suite du rachat par Googleen 2005 de la startup du même nom[4]. En 2018, Android est le système d'exploitation le plus utilisé dans le monde avec plus de 80 % de parts de marché dans les smartphones.

- **Pourquoi avoir choisi Android ?**

La domination d'Android sur le marché des smartphones est incontestable. Android présente plus de 84,1% des terminaux livrés au cours du trimestre ou 293,7 millions d'unités comme le montre la figure num…

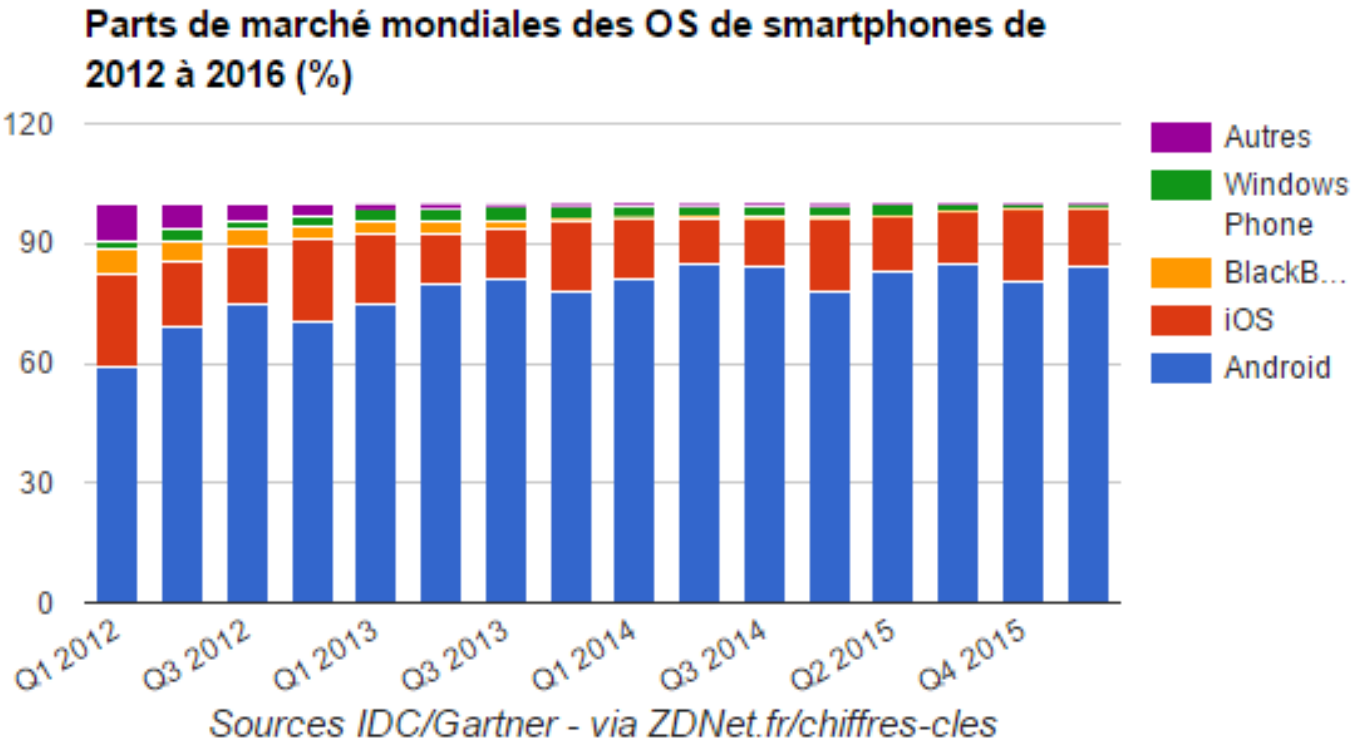
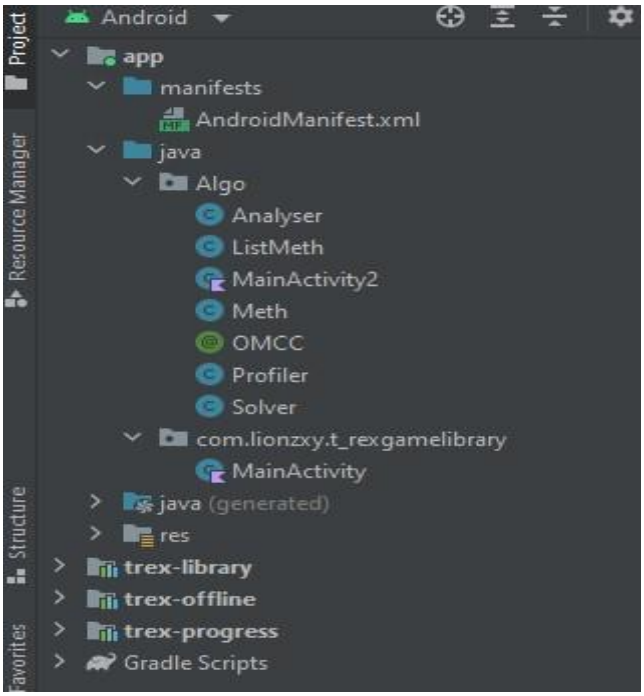


Fig : Parts de marché mondiales des SEs de smartphone (2012-2016)

1.4.2 L'architecture du projet

Pour l'implémentation de notre approche nous avons respecté l'architecture proposée dans le chapitre 3, nous avons commencé par développer les deux premiers modules profileur et analyseur pour utiliser ses résultats dans l'implémentation de dernier module solveur.



❖ Profileur

Dans cette partie nous présentons des portions de code source de chaque module avec l'explication nécessaire

- ProfileurMatériel

Code pour connaitre le pourcentage de batterie actuel nous utilisons un broadcastReceiver Pour pouvoir recevoir des intents, Android permet de créer une classe qui implémente BroadcastReceiver. Ces objets sont conçus pour recevoir des intents (intentions) et appliquer des comportements spécifiques. L'interfaceBroadcastReceiverne possèdequ'uneseuleméthodeonReceive().

```
public BroadcastReceiver batteryInfos(){

    BroadcastReceiver   br = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            double level = intent.getIntExtra(BatteryManager.EXTRA_LEVEL,  defaultValue: 0);
            Log.d( tag: "batri",  msg: "batteryPct:"+level);
        }
    } ;
    return br;
}
```

- Code pour récupérer la capacité CPU

```
public long CPUcapacity(){
    try{
        reader = new BufferedReader(new InputStreamReader(new FileInputStream( name: "/proc/stat")), sz: 1000);
    } catch(FileNotFoundException e){
        e.printStackTrace();
    }
    String load = null;
    try {
        load = reader.readLine();
    } catch (IOException e){
        e.printStackTrace();
    }
    try{
        reader.close();
    }   catch (IOException e){
        e.printStackTrace();
    }
    String[] toks = load.split( regex: " ");
    long currTotal = Long.parseLong(toks[2])+Long.parseLong(toks[3])+Long.parseLong(toks[4]);
    long currIdle = Long.parseLong(toks[5]);
    long total=0, idle=0;
    Usage = (currTotal-total)*100.0f / (currTotal-total + currIdle - idle);
    total = currTotal;
    idle = currIdle;
    return idle;
```

Project update recommended

- Code pour avoir la RAM disponible

```java
public double RAMcapacity(){
    try {
        RandomAccessFile reader = new RandomAccessFile( name: "/proc/meminfo", mode: "r");
        String load = reader.readLine();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
    Pattern p= Pattern.compile("(\\d+)");
    CharSequence load = null;
    Matcher m=p.matcher(load);
    String value = "" ;
    while(m.find()){
        value = m.group(1) ;
    }
    try {
        reader.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    double totRam = Double.parseDouble(value);
    double mb = totRam / 1024.0;

    return mb ;
}
```

- **Profileur Réseau :**

  - Code pour connaitre la disponibilité du réseau et internet :

```
public static boolean isConnectedToInernet(Context c){
    ConnectivityManager cm = (ConnectivityManager) c.getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo ni = cm.getActiveNetworkInfo() ;
    return ((ni != null)&&(ni.isConnected())) ;
}

public static boolean isNetworkAvailable(Context context){
    ConnectivityManager connectivity = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE
    if(connectivity==null){
        return false ;
    } else {
        NetworkInfo[] info = connectivity.getAllNetworkInfo() ;
        if(info != null) {
            for (int i = 0 ; i< info.length ; i++) {
                if (info[i].getState()== NetworkInfo.State.CONNECTED){
                    return true ;
                }
            }
        }
    }
    return false ;
}
```

- Code pour connaitre le type de réseau utilisé (4G/wifi) :

```
public static final String connectionType(Context context){
    final ConnectivityManager connMgr = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
    final NetworkInfo wifi = connMgr.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
    final NetworkInfo mobile = connMgr.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
    if (wifi.isAvailable() && wifi.getDetailedState() == NetworkInfo.DetailedState.CONNECTED){
        return "4G";
    }
    return null ;
}
```

- Code pour connaitre la qualité du debit :

```
public boolean isConnectionFast (Context context) {
    ConnectivityManager cm = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo net = cm.getActiveNetworkInfo();
    if (net.getType() == ConnectivityManager.TYPE_WIFI) {
        return true;
    } else if (net.getType() == ConnectivityManager.TYPE_MOBILE) {
        switch (net.getSubtype()) {
            case TelephonyManager.NETWORK_TYPE_1xRTT:
                return false;
            case TelephonyManager.NETWORK_TYPE_CDMA:
                return false;
            case TelephonyManager.NETWORK_TYPE_EVDO_0:
                return true;
            case TelephonyManager.NETWORK_TYPE_EDGE:
                return false;
            case TelephonyManager.NETWORK_TYPE_EVDO_A:
                return true;
            case TelephonyManager.NETWORK_TYPE_GPRS:
                return true;
            case TelephonyManager.NETWORK_TYPE_HSDPA:
                return true;
            case TelephonyManager.NETWORK_TYPE_HSPA:
                return true;
            case TelephonyManager.NETWORK_TYPE_HSUPA:
                return true;
            case TelephonyManager.NETWORK_TYPE_UMTS:
```

Project update recommen

- **Profileur Logiciel**

  - Code pour récupérer la méthode Actuelle

```
@OMCC
public static String getMethodName(final int depth){
    final StackTraceElement[] ste = Thread.currentThread().getStackTrace();
    return ste[ste.length-1-depth].getMethodName() ;
    Method cm =;
    String nameCurrent;
    for(Method m : MainActivity.class.getMethods()){

        if (!m.getName().equals(nameCurrent)) {
            continue;
        }
        cm = m;
    }
}

public static void getMethod(){
    Throwable t = new Throwable() ;
    StackTraceElement[] elements = t.getStackTrace() ;

    String callerMethod = elements[0].getMethodName();
    String callerMethodeName = elements[1].getMethodName();
    String callerClassName = elements[1].getClassName() ;
    Log.e( tag: "resultats", msg: "CallerClassName="+callerClassName+", Caller method name:"+ callerMethodeName);
}
```

❖ **Analyseur :**

Représente une analyse de programme de formulaire pour identifier les relations de dépendance entre les composants d'une application mobile.

- ce code pour analyser chaque méthode et choisir celles qui peuvent être chargées dans le cloud, nous utilisons des listes chainées pour stocker toutes les données de méthode.

```
try {
    while(m!=null){
        Log.d( tag: "methodes",  msg: "entering while ");

        if(!m.conf){
            Log.d( tag: "methodes",  msg: "if(!m.conf)");

            cloudL.idMethode = i ;
            i++;
            cloudL.nameMethode = m.nameMethode ;
            cloudL.cloudAccess = m.cloudAccess ;
            cloudL.mobileAccess = m.mobileAccess ;
            cloudL.conf = m.conf ;

            Log.d( tag: "methodes",  msg: "Id"+cloudL.idMethode + "  "+cloudL.nameMethode +
                    "  "+cloudL.conf+ "  "+cloudL.cloudAccess+"  "+cloudL.mobileAccess+"\n");

            ctail.setNext(cloudL);
            ctail = cloudL ;
            cloudL = new Meth() ;
            //m=m.getNext() ;

        }
```

le processus d'analyse nous a donné deux listes distinctes, une pour le cloud et une pour le mobile.

Mobile liste :

```
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: this is mobile list
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id1  main:processDebug  false  0.41685869070472836  0.46685869070472835
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id2  TRexOfflineActivity:creat  false  0.5008850766638036  0.5508850766638036
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id3  TRexOfflineView  false  0.6210390692415341  0.6710390692415341
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id4  TRexGameView:start  false  0.37679988166794676  0.42679988166794675
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id5  TRexOfflineView:running  false  0.8515595290474052  0.9015595290474052
2022-07-06 18:46:23.085 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id6  TRexOfflineView:fault  false  0.08886496390566057  0.13886496390566055
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id7  IErrorListner  false  0.4961126103215344  0.5461126103215135
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id8  IProgressListener  false  0.19518874298224598  0.24518874298224597
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id9  ITRexZipProvider  false  0.8292236286913905  0.8792236286913906
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id10  TRexPlayActivity  false  0.40987021877283814  0.4598702187728381
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id11  TRexProgressView  false  0.15621050936080416  0.20621050936080415
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id12  ITRexGameView  false  0.10517453353026873  0.15517453353026872
2022-07-06 18:46:23.086 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id13  showProgress  false  0.3198052014855913  0.3698052014855591
2022-07-06 18:46:23.087 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id14  Analyser:getData  false  0.40129778129741145  0.45129778129741144
2022-07-06 18:46:23.088 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id15  Analyser:CPUProcess  true  0.976270998145421  1.026270998145421
2022-07-06 18:46:23.088 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id16  Analyser:Ram  true  0.5560303743197127  0.6060303743197127
2022-07-06 18:46:23.088 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id17  Profiling:dataMeth  true  0.4448259509314678  0.4948259509314678
2022-07-06 18:46:23.088 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id18  Profiling:fault  true  0.3089232171638958  0.3589232171638958
2022-07-06 18:46:23.088 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id19  Solving:ProcessDebug  true  0.15055614071240475  0.20055614071240474
```

Cloud liste :

```
2022-07-06 18:46:23.082 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id19  Solving:ProcessDebug  true  0.15055614071240475  0.20055614071240474
2022-07-06 18:46:23.082 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: this is cloud list
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id1  main:processDebug  false  0.41685869070472836  0.46685869070472835
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id2  TRexOfflineActivity:creat  false  0.5008850766638036  0.5508850766638036
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id3  TRexOfflineView  false  0.6210390692415341  0.6710390692415341
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id4  TRexGameView:start  false  0.37679988166794676  0.42679988166794675
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id5  TRexOfflineView:running  false  0.8515595290474052  0.9015595290474052
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id6  TRexOfflineView:fault  false  0.08886496390566057  0.13886496390566055
2022-07-06 18:46:23.083 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id7  IErrorListner  false  0.49611261032151344  0.5461126103215135
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id8  IProgressListener  false  0.19518874298224598  0.24518874298224597
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id9  ITRexZipProvider  false  0.8292236286913905  0.8792236286913906
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id10  TRexPlayActivity  false  0.40987021877283814  0.4598702187728381
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id11  TRexProgressView  false  0.15621050936080416  0.20621050936080415
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id12  ITRexGameView  false  0.10517453353026873  0.15517453353026872
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id13  showProgress  false  0.31980520148555913  0.3698052014855591
2022-07-06 18:46:23.084 9323-9323/com.lionzxy.t_rexgamelibrary D/methodes: Id14  Analyser:getData  false  0.40129778129741145  0.45129778129741144
```

- **Solveur :**

après avoir rassemblé les deux listes de l'analyseur et collecté toutes les données du profileur, l'algorithme du solveur peut maintenant être exécuté et nous a donné la meilleure décision de partitionnement et l'optimisation de l'utilisation de la batterie à chaque exécution du code.

```
public void Algrm(){
    int i = 0;

    while(i<loop){
        Log.d( tag: "methodes",  msg: "this is main method");

        Meth m = new Meth();
        ListMeth l =new ListMeth();
        m = l.creatList() ;
        l.readListe2(m);
        Log.d( tag: "methodes",  msg: "calling analyse class after creation");
        Analyser n = new Analyser(l);
        n.analyse(m);
        Log.d( tag: "methodes",  msg: "this is cloud list\n");
        l.readListe2(n.getCloudL());
        Log.d( tag: "methodes",  msg: "this is mobile list\n");

        l.readListe2(n.getMobileL());
        Profiler p=new Profiler();
        this.mReceiver = p.batteryInfos();

        i++;
    }
}
```

la capture suivante est la pure différence de niveau de batterie dans plusieurs cas d'exécution du jeu T-Rex.

```
2022-07-06 19:11:18.966 9741-9741/com.lionzxy.t_rexgamelibrary D/battery: batteryPct:98
2022-07-06 19:12:47.098 9879-9879/com.lionzxy.t_rexgamelibrary D/battery: batteryPct:97.45
2022-07-06 19:18:00.091 10028-10028/com.lionzxy.t_rexgamelibrary D/battery: batteryPct:99.0
2022-07-06 19:18:00.091 10028-10028/com.lionzxy.t_rexgamelibrary D/battery: batteryPct:97.14090463795056
2022-07-06 19:18:00.091 10028-10028/com.lionzxy.t_rexgamelibrary D/battery: batteryPct:97.45
2022-07-06 19:21:09.950 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: le niveau d batterie actuel:
2022-07-06 19:21:09.950 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: intent.getIntExtra(BatteryManager):99.0
2022-07-06 19:21:09.950 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: le niveau d batterie Mobile exe:
2022-07-06 19:21:09.951 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: intent.getIntExtra(BatteryManager):96.63834466420916
2022-07-06 19:21:09.951 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: le niveau d batterie Cloud:
2022-07-06 19:21:09.951 10152-10152/com.lionzxy.t_rexgamelibrary D/battery: intent.getIntExtra(BatteryManager):97.45
```

Il montre dans la vue des résultats que le niveau de la batterie après l'exécution de l'application dans le mobile perce beaucoup d'énergie que lorsqu'il s'exécute dans le cloud,

Consommation mobile --> 99,0-96,63 = 2,37 / presque consommé 2,39%

Consommation cloud --> 99,0-97,45 = 1,55 / presque consommé 1,56 %

On calcule maintenant la différence

2,39 - 1,56 = 0,38

Donc l'énergie économisée en utilisant notre algorithme est de près de 0,38% de la consommation totale, et c'est une bonne valeur.

1.1 Conclusion Generale Et Perspectives

Ce mémoire présente l'ensemble des travaux réalisés dans le cadre de notre projet de fin d'études. Nous avons passé en revue l'état de l'art et étudié la littérature scientifique concernant les domaines de recherche abordés. Nous avons introduit le concept du Mobile Cloud Computing en étudiant les standards et travaux de recherches existants et proposés, tout en mettant l'accent sur les travaux qui traitent la consommation d'énergie en particulier.
Nos études nous ont permis d'identifier les limites et manques au niveau des travaux connexes choisi, dirigeât nos travaux de recherche et nous motivant à essayer d'apporter une contribution dans le but de proposer des solutions adéquates à ces limites.
Nos travaux nous ont menées à notre contribution : Un algorithme basé sur une boucle qui optimise l'énergie jusqu'à 0.38%, set algorithme rassemble toutes les données et le temps d'accès dans la mémoire, et le nombre d'itérations est l'indice d'une meilleure optimisation.

avec l'aide de notre professeur, nous avons développé cet algorithme simple, et il nous a donné les résultats souhaités, le point fort de notre algorithme est sa simplicité, tout ce dont nous avions besoin était de rassembler toutes les données et informations nécessaires sur l'application, puis nous l'avons juste donnée à l'algorithme, et cela nous a donné une meilleure solution.

Cette expérience fut très bénéfique pour nous, elle nous a permis d'acquérir plusieurs compétences d'un point de vue analyse, synthèse et programmation. Nous espérons que tous ces concepts seront utiles dans notre vie professionnelle.

1 Conclusion générale

1.1 Synthèse

Ces dernières années, la demande croissante de l''informatique mobile, qui permet aux gens d'être connectés en permanence à Internet, a radicalement changé la façon dont les gens font les choses.La déportation de calculs dans le Cloud a été proposée pour résoudre les problèmes de performance sur le terminal mobile

Dans Mobile Cloud Computing, le déchargement de calculs devient une méthode prometteuse pour réduire le temps d'exécution des tâches et prolonger la durée de vie de la batterie des appareils mobiles. L'idée principale du déchargement de calculs est de migrer les calculs lourds des appareils mobiles vers des serveurs cloud.

Ce mémoire présente l'ensemble des travaux réalisés dans le cadre de notre projet de fin d'études. Nous avons passé en revue l'état de l'art et étudié la littérature scientifique concernant les domaines de recherche abordés. Nous avons introduit le concept du Mobile Cloud Computing en étudiant les standards et travaux de recherches existants et proposés, tout en mettant l'accent sur les travaux qui traitent la consommation d'énergie en particulier.

Pour résoudre le problème de partitionnement des applications dans un environnement mobile, nous avons proposé un nouvel algorithme de partitionnement qui trouve le partitionnement optimal des applications sous différents modèles de coûts et parvient aux meilleurs compromis entre l'économie de temps/énergie. Pour ce faire, il construit des graphes d'appel pondérés pour les applications.

Les résultats expérimentaux montrent que selon de l'environnement (par exemple, la bande passante et la vitesse du serveur), l'algorithme proposé peut atteindre résultat de partitionnement optimal en termes d'économie de temps et d'énergie. Le déchargement profite beaucoup des bandes passantes élevées et des facteurs d'accélération importants, tandis que les bandes passantes basses favorisent le schéma de non déchargement.

Enfin nous pouvons dire que ce projet de fin d'étude nous a permis d'aborder plusieurs aspects de l'informatique tels que : Le Cloud Computing, Mobile Computing, Mobile Cloud Computing, déchragment de caluls et les algorithmes de partitionnement.

Cette expérience fut très bénéfique pour nous, elle nous a permis d'acquérir plusieurs compétences d'un point de vue analyse, synthèse et programmation. Nous espérons que tous ces concepts seront utiles dans notre vie professionnelle.

1.2 Perspectives

Vu que dans le domaine d'informatique l'existence d'un travail parfait et sans limitations est impossible, et comme c'est le cas pour notre approche et implémentation, d'autres travaux peuvent être réalisés dans le but de l'améliorer :

L'algorithme utilisé peut être amélioré par l'utilisation d'une métaheureustique ;
Améliorer la sécurité dans l'offloading