進捗報告資料

安達智哉
to-adachi@ist.osaka-u.ac.jp

2019 年 9 月 25 日

1　CQ 研究会

論文を執筆中である。

2　Idle タイマの最適化

以前までの評価において、Idle タイマを適切に設定することにより、CPU 負荷およびメモリ使用量の削減が期待できることを示した。それと同時に、Idle タイマの最適な値は、UE の通信周期やその分布に依存して大きく変化することも示した。

一方で、現実的には UE の通信周期やその分布は明らかではない。また、それらは時間とともに変化するものである。そのため、UE の通信周期やその分布が不明であり、動的に変化するような環境においても、Idle タイマを適切な値に設定するような制御方法が必要となる。そこで本章では、Idle タイマの制御方法に関して述べる。第 2.1 節で Idle タイマの更新タイミングについて述べる。第 2.2 節で Idle タイマを制御する上での目的関数を定義する。第 2.3 節で目的関数を最小化するための Idle タイマの制御方法を述べる。

2.1　Idle タイマの更新タイミング

MME は、自身の負荷に応じて Idle タイマを適応的に変化させる。この時、MME が決定した Idle タイマを各 UE に適用するタイミングは、複数通り考えられる。まず、UE の強制的な状態遷移を引き起こさない方法として以下の 2 つが考えられる。

- UE がアタッチしたタイミング
- UE がデータ送信を行うタイミング

次に、UE の強制的な状態遷移を引き起こす方法として以下の 2 つが考えられる。

- UE の動作や状態に依存しない、定期的なタイミング
- 任意のタイミング

それぞれには、メリットデメリットが考えられる。

UE の強制的な状態遷移を引き起こさない方法の場合、更新タイミングが UE の動作に依存するため、新しい Idle タイマを UE に設定するまでにかかる時間が UE ごとに異なるという問題がある。これにより、異なる Idle タイマを持つ UE が同時に存在するような状況が発生するため、Idle タイマの制御が複雑になると考えられる。また、Idle タイマを変更した後、CPU 負荷とメモリ使用量に変化が現れるまでに遅延が発生するため、MME のリソースの制御が難しくなると考えられる。しかし、アタッチやデータ送信など、UE と MME が通信するタイミングで Idle タイマの更新を行うため、追加のシグナリングや状態遷移が少なく、オーバヘッドが小さい。

一方、更新タイミングが UE の動作や状態に依存しない場合、Idle タイマを更新するために UE の状態を変化させる必要がある場合があり、オーバヘッドが大きくなるという問題がある。具体的には、MME と通信できない状態にある UE の Idle タイマを変化させるためには、UE を一度接続状態へと遷移させる必要がある。この際に、状態遷移に伴うシグナリング処理が発生するため、MME の CPU 負荷が増加する。しかし、Idle タイマを UE に反映させるまでにかかる時間は UE に依存しないため、全 UE の Idle タイマの値を一定期間内で更新できる。さらに、設定する Idle タイマの値を 0 にすることで、MME は任意のタイミングで任意の UE を強制的にアイドル状態へ遷移させることができる。これにより、UE の強制的な状態遷移を引き起こさない方法の場合と比較して MME のリソース制御が容易になると考えられる。

2.2 目的関数の定義

本節では、UE の強制的な状態変化を引き起こさないことを前提にする。つまり、Idle タイマが切れていない UE を強制的に Idle 状態へ遷移させることなないとする。また、Idle タイマの更新は、UE がデータ送信を行うタイミングで実行するものとする。MME は UE を収容するために使用されている CPU およびメモリリソース量を観測できるものとする。つまり、UE の収容とは無関係な処理によって発生する負荷を取り除いた CPU 負荷およびメモリ使用量を知ることができるとする。MME は現在収容されている UE 台数を観測できるものとする。

突発的な負荷の増加に対応するという観点から、現在収容している UE に加え、最も多くの UE を収容できるような Idle タイマの値が最適と考える。具体的には、現在収容している UE と同じ通信周期を持つ UE がネットワークに参加すると仮定し、最も多くの UE を追加で収容できる Idle タイマの値を最適と定義する。また、CPU よびメモリのどちらも過負荷状態でないことは、UE を収容可能であることの必要十分条件であるとする。

まず、UE 一台あたりが各リソースに与える負荷の平均を推定する。現在収容している UE 台数を N_{UE} とする。UE 台数が N_{UE}、Idle タイマが T^{i} の時に観測される、CPU 負荷およびメモリ使用量をそれぞれ $C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$、$M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$ とする。この時、UE 一台あたりが与える CPU 負荷およびメモリ使用量の平均 ($C_1(T^{\mathrm{i}})$、$M_1(T^{\mathrm{i}})$) は以下の式 (1)、(2) で表せる。

$$C_1(T^{\mathrm{i}}) = \frac{C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{N_{\mathrm{UE}}} \tag{1}$$

$$M_1(T^{\mathrm{i}}) = \frac{M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{N_{\mathrm{UE}}} \tag{2}$$

Idle タイマを T^{i} とした時に、N_{UE} 台の UE を収容している状態から追加で収容可能な UE 台数を $N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ とする。$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ は、$C_1(T^{\mathrm{i}})$、$M_1(T^{\mathrm{i}})$、$C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$、$M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$、$C^{\max}$ および $M^{\max}$ を用いて、以下の式 (3) で表せる。ここで、$C^{\max}$，$M^{\max}$ はそれぞれシグナリング処理および UE

のセッション情報を保持するために使用可能な CPU リソース量およびメモリリソース量である。

$$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}}) = \quad \min\{\lfloor\frac{C^{\max}-C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{C_1(T^{\mathrm{i}})}\rfloor, \lfloor\frac{M^{\max}-M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{M_1(T^{\mathrm{i}})}\rfloor\} \tag{3}$$

Idle タイマを制御する上での目的関数を以下の式 (4) に示す。

$$\text{maximize}: \quad N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}}) \tag{4}$$

2.3 Idle タイマの制御方法

$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ を最大化するように、Idle タイマを制御すれば、Idle タイマを最適な値に近づけることができる。具体的には、各リソースの使用量を観測して、$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ を大きくする向きに Idle タイマを変化させる。このステップを複数回繰り返すことにより、Idle タイマを制御する。

この時、1 ステップごとの Idle タイマの変化量を考える必要がある。この値を小さく設定すると、最適な値に到達するまでに大きな時間がかかってしまう場合がある。逆に Idle タイマの変化量を大きく設定すると、Idle タイマが発振する可能性もあり、制御が不安定になる。また、UE の通信周期によって、Idle タイマが変化した時に各リソースの負荷の変化量が異なる点も考慮する必要がある。

つまり、ネットワークの変化に短い時間スケールで対応しつつ、安定した制御を実現するためには、ネットワークの環境に応じて Idle タイマの変化量を制御する仕組みが必要である。この仕組みには PID 制御が利用できると考えている。Idle タイマの設定を入力、各リソースの使用率を出力をして捉えることで、PID 制御により、Idle タイマの変化量を調整しつつ、最適値に近づけることができる。

以前の評価より、UE 台数を固定した時、CPU 負荷は Idle タイマの値に対して広義単調減少でありかつ、メモリ使用量は Idle タイマの値に対して広義単調増加であることがわかっている。このことから、$C_1(T^{\mathrm{i}})$ および $C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$ は T^{i} に対して広義単調減少であることがわかる。同様に $M_1(T^{\mathrm{i}})$ および $M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})$ は T^{i} に対して広義単調増加であることがわかる。以上を踏まえて式 (3) を確認すると、$\lfloor\frac{C^{\max}-C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{C_1(T^{\mathrm{i}})}\rfloor$ は広義単調増加でありかつ、$\lfloor\frac{M^{\max}-M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{M_1(T^{\mathrm{i}})}\rfloor$ は広義単調減少であることがわかる。ここで、$|\lfloor\frac{C^{\max}-C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{C_1(T^{\mathrm{i}})}\rfloor$ と $\lfloor\frac{M^{\max}-M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{M_1(T^{\mathrm{i}})}\rfloor|$ の差分を最小化するような T^{i} の集合を $\boldsymbol{T^{\mathrm{i}}}$ とする。また、$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ を最大化するような T^{i} の集合を $\boldsymbol{T^{\mathrm{i}}_{\mathrm{optimal}}}$ とする。すると、$\lfloor\frac{C^{\max}-C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{C_1(T^{\mathrm{i}})}\rfloor$ は広義単調増加でありかつ、$\lfloor\frac{M^{\max}-M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{M_1(T^{\mathrm{i}})}\rfloor$ は広義単調減少であることを考慮すると、$T^{\mathrm{i}} \in \boldsymbol{T^{\mathrm{i}}}$ であることは $T^{\mathrm{i}} \in \boldsymbol{T^{\mathrm{i}}_{\mathrm{optimal}}}$ であるための十分条件になる。

このことを踏まえ、PID 制御における出力値 $y(t)$ および目標値 $r(t)$ を以下の式 (5)、(6) のように定義する。t は時刻を表す変数である。

$$y(t) \quad = \quad \lfloor\frac{C^{\max}-C_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{C_1(T^{\mathrm{i}})}\rfloor - \lfloor\frac{M^{\max}-M_{N_{\mathrm{UE}}}(T^{\mathrm{i}})}{M_1(T^{\mathrm{i}})}\rfloor \tag{5}$$

$$r(t) \quad = \quad 0 \tag{6}$$

(※ $N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ を最大化することが本来の目的であるが、最大化問題はそのままでは PID 制御に落とし込めないと考え、式 (5)、(6) に示すように、$N_{\mathrm{UE}}^{\mathrm{add}}(T^{\mathrm{i}})$ を用いない形で、出力値および目標値を定義した。本節の第 4 段落で述べている通り、目的関数は異なるが得られる結果は正しい値であると思われる。)

3　今後の予定

- PID 制御に関する学習
- 論文の執筆
  - 完成: 〜 9/25
- 発表スライドの作成: 〜10/16