進捗報告資料

安達智哉
to-adachi@ist.osaka-u.ac.jp

2019 年 9 月 11 日

1　CQ 研究会

論文を執筆中である。

2　Idle Timer の最適化

以前までの評価において、Idle Timer を適切に設定することにより、CPU およびメモリのリソース使用率の削減が期待できることを示した。それと同時に、Idle Timer 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 Timer 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 Timer の制御方法に関して述べる。まず、第 2.1 節で Idle Timer を制御する上での目的関数を定義する。そして第 2.2 節で目的関数を最小化するための Idle Timer の制御方法を述べる。

2.1　目的関数の定義

CPU リソースの使用率およびメモリリソースの使用率をそれぞれ U_{cpu} および U_{memory} と定義する。U_{cpu} および U_{memory} がどちらも小さい値を取るような Idle Timer の設定が MME にとって良いことは明らかであるが、U_{cpu} と U_{memory} の間にはトレードオフの関係があるため、U_{cpu} と U_{memory} の双方を最小化させることは一般的に困難である。

一方で、突発的な負荷の増加に対応するという観点から、最も余力のある状態が Idle Timer の最適値とする考え方がある。そのような最適化を目標とする場合、U_{cpu} および U_{memory} がどのような時に、最も余力のある状態と言えるのかを考える必要がある。まず、突発的な負荷は、現在のネットワークに存在する UE と同じような通信周期を持つ UE が大量にネットワークに参加することにより発生すると仮定する。この仮定に基づくと以前の評価結果より、U_{cpu} および U_{memory} は共に UE 台数に対して比例して増加することになる。今回の条件では、CPU およびメモリのどちらか一方でも過負荷状態になると、UE の収容が難しくなるため、使用率の高い方のリソースがボトルネックになることに注意する。

上述のような点を考慮した、目的関数の一案として以下の式 (1) を示す。目的関数 f は CPU とメモリのうち、使用率が高い方の値を取る。この目的関数 (f) を最小化するような Idle Timer の値は、ボトルネックとなるリソースの負荷を緩和するという意味では最適であると考えられる。

$$f = \max\{U_{\mathrm{cpu}}, U_{\mathrm{memory}}\} \tag{1}$$

さらに、CPU 負荷は Idle Timer の値に対して広義単調減少 (単調非増加) でありかつ、メモリ負荷は Idle Timer の値に対して広義単調増加 (単調非減少) であるという特徴があるため、目的関数 f を以下の式 (2) のように書き換えることも可能である。式 (2) では、CPU 使用率とメモリ使用率が均等化するような Idle Timer の値が最適であるとしている。グラフ上で見ると、(0,0) と (メモリ負荷の最大値,CPU 負荷の最大値) とを結ぶ直線との交点を示す。上述の特徴のため、式 (2) から求められる Idle Timer の値は式 (1) から求められる Idle Timer の値の部分集合となる。

$$f = \min\{|U_{\mathrm{cpu}} - U_{\mathrm{memory}}|\} \tag{2}$$

2.2 Idle Timer の制御方法

今回の評価では、CPU 負荷は Idle Timer の値に対して広義単調減少 (単調非増加) でありかつ、メモリ負荷は Idle Timer の値に対して広義単調増加 (単調非減少) であるため、CPU 負荷を削減する場合には Idle Timer を増加させ、メモリ負荷を削減する場合はその逆の操作を行えば良い。このことを考慮しつつ、目的関数 (f) を最小化するように、Idle Timer を制御すれば、Idle Timer を最適な値に近づけることができる。具体的には、リソースの使用率を観測して、f を小さくする向きに Idle Timer 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 Timer を制御する。

この時、1 ステップごとの Idle Timer 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 Timer の変化量を大きく設定すると、Idle Timer が発振する可能性もあり、制御が不安定になる。また、UE の分布によって、Idle Timer を変化がリソース負荷へ与える影響が変化す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 Timer 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 Timer 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 Timer の変化量を調整しつつ、最適値に近づけることができる。

3 今後の予定

- 論文の執筆
  - 5,6 章: 〜 9/18
  - 完成: 〜 9/25
- 発表スライドの作成: 〜10/16