Preferred Networks インターン選考2019 コーディング課題 バックエンド分野

変更履歴

- 2019年4月19日 : 初版

回答にあたっての注意

- ソースコードには以下のいずれかの言語を使用してください。
  - C, C++, Python, Go, Scala, Rust, Java, C#, Ruby
  - 複数の言語を使用しても構いません。
- 各言語の標準ライブラリの関数のみを使用してください。
  - ただし、ロギング処理およびサーバー・ワーカー間通信処理のみ、外部ライブラリを使用しても構いません。
- 課題には自分だけで取り組んでください。この課題を他の応募者を含めた他人と共有・相談することを禁止します。課題期間中に GitHub の公開リポジトリ等にソースコードや問題をアップロードする行為も禁止します。漏洩の証拠が見つかった場合、その応募者は失格となります。ある応募者が別の応募者に回答をコピーさせた場合、双方の応募者が失格となります。
- 想定所要時間は最大2日です。全課題が解けていなくても提出は可能ですので、学業に支障の無い範囲で取り組んで下さい。

提出物

- 課題のプログラムのソースコード
- 実行方法を示したレポート
  - プログラム開発に使用したOS・プログラミング言語・バージョンを記述してください
  - プログラムのビルドや、実行に必要なMakefile等のコマンドを添付・記述してください

評価基準

以下のような要素を評価します。

- 出力された結果が正しいこと。
- 実行速度が速いこと。
- 使用メモリ量が少ないこと。
- ソースコードが他人が読んで理解しやすいものになっていること。
- ソースコードが正しいものであることを検証するために、単体テストや検証のためのコードが書かれていること。
- ライブラリを適切に選定・使用しているか。
- 提出物を見て追試がしやすい形になっていること。
- レポートが要点についてわかりやすくまとまっていること。

一次選考後の面接で提出されたコードについて質問する可能性があります。

提出方法

上記の提出物を単一のパスワード無しzipファイルにまとめ、こちらの専用フォームより応募してください。締切は日本時間2019年5月7日（火）12:00です。

問い合わせ

問題文に関して質問がある場合はintern2019-admin@preferred.jpまでご連絡ください。問題文に訂正が行われた場合は応募者全員にアナウンスいたします。なお、アプローチや解法に関する問い合わせにはお答えできません。

問題文

以下の2つから成るジョブ実行システムを作成します。

1. 与えられた時刻に対してジョブを返すサーバー
2. 取得したジョブに記載されている複数のタスクを順番に実行するワーカー

サーバーが返すジョブのリストは、課題ディレクトリに含まれる以下のデータを対象とします。

- job_list.zip (MD5: `d2f84ff78eced143c1c12780155be267`)

データには `xxxx.job` というファイルがあり、以下の項目が含まれています。

```
data/
├── 00001.job
├── 00002.job
├── 00003.job
├── ...
```

1. `JobID`: ジョブID
2. `Created`: ジョブが作成された時刻
3. `Priority`: "High" および "Low" いずれかの優先度。
4. `Tasks`: 実行ポイント列 (1つのジョブに対して複数のタスクからなり、ポイントは各タスクの実行時間と一致しています)

前提条件

- 同一JobIDのジョブのタスクは必ず順番に実行すること。並行して実行はできない。
- JobIDの異なるジョブは並行して実行してよい。
- タスクとタスクの間に待ちは発生してよいが、タスク中に待ちを挿入することはできない。例えば、`Tasks=[2,2]` の場合、最初の2ポイントと次の2ポイントの間に待ちは発生してよいが、2ポイントを1と1に分割して間に待ちを挿入することはできない。
- サーバーとワーカー間の通信コストは考慮しない。また、時刻は同期しているとする。
- JobIDは全体で一意なID
- 時刻の最小単位は秒

問題1-1

与えられた時刻に対して、作成時刻がその時刻となっているジョブを返すウェブサーバーを作成してください。APIは適切に決定してください。

例えば、以下の2つのジョブがサーバーに登録されているとします。

00001.job

1. `JobID`: 1
2. `Created`: 00:00:01
3. `Priority`: Low
4. `Tasks`: 5, 6, 7

00002.job

1. `JobID`: 2
2. `Created`: 00:00:03
3. `Priority`: High
4. `Tasks`: 3, 5

この場合は以下の仕様となります。

- `time=00:00:01` --> 00001.job の内容を返す
- `time=00:00:02` --> 「なし」を返す
- `time=00:00:03` --> 00002.job の内容を返す

問題**1-2**

問題1-1で作成したサーバーに対して問い合わせを行い、取得したジョブに記載のタスクを実行するワーカーを作成してください。各時刻での実行中の総ポイントの推移をグラフにし、提出物に含めてください。

タスクの実行ポイント列は実行にかかる時間を表し、1ポイントは1秒を表しています。ワーカーはタスクを開始してから、1秒ごとにタスクにあるポイントを1ずつ消費していきます。 実際に**1**秒間の**sleep**処理を入れる必要はありません。

問題1-1の例だと、以下のような実行スケジュールとなります。実行中のポイント推移は `Executing Point` に該当します。

timestamp	`JobID`-`Task No`(`Remain Point`)		Executing Point
00:00:01	1-1(5)		5
00:00:02	1-1(4)		4
00:00:03	1-1(3)	2-1(3)	6
00:00:04	1-1(2)	2-1(2)	4
00:00:05	1-1(1)	2-1(1)	2
00:00:06	1-2(6)	2-2(5)	11
00:00:07	1-2(5)	2-2(4)	9
00:00:08	1-2(4)	2-2(3)	7
00:00:09	1-2(3)	2-2(2)	5
00:00:10	1-2(2)	2-2(1)	3
00:00:11	1-2(1)		1
00:00:12	1-3(7)		7
...			
00:00:18	1-3(1)		1

問題**2-1**

問題1-2で作成したワーカーに、実行ポイントの上限 (以下"キャパシティ") を設定できるようにしてください。キャパシティを15に設定し、実行中のポイント推移をグラフ化して提出物に含めてください。

1つのタスクの実行ポイントがキャパシティを超えることはないとします。

問題1の例で、キャパシティを10とした場合は以下のような実行スケジュールとなります。

timestamp	`JobID`-`Task No`(`Remain Point`)		Executing Point
00:00:01	1-1(5)		5
00:00:02	1-1(4)		4
00:00:03	1-1(3)	2-1(3)	6
00:00:04	1-1(2)	2-1(2)	4
00:00:05	1-1(1)	2-1(1)	2
00:00:06	1-2(6)		6
00:00:07	1-2(5)	2-2(5)	10
00:00:08	1-2(4)	2-2(4)	8
00:00:09	1-2(3)	2-2(3)	6
00:00:10	1-2(2)	2-2(2)	4
00:00:11	1-2(1)	2-1(1)	2
00:00:12	1-3(7)		7
…			
00:00:18	1-3(1)		1

問題**2-2**

問題2-1で設定したキャパシティに加えて、優先度を考慮して実行できるようにしてください。同じく、キャパシティを15に設定し、実行中のポイント推移をグラフ化して提出物に含めてください。

timestamp	`JobID`-`Task No` (`Remain Point`)		Executing Point
00:00:01	1-1(5)		5
00:00:02	1-1(4)		4
00:00:03	1-1(3)	2-1(3)	6
00:00:04	1-1(2)	2-1(2)	4
00:00:05	1-1(1)	2-1(1)	2
00:00:06		2-2(5)	6
00:00:07	1-2(6)	2-2(4)	10
00:00:08	1-2(5)	2-2(3)	8
00:00:09	1-2(4)	2-2(2)	6
00:00:10	1-2(3)	2-2(1)	4
00:00:11	1-2(2)		2
00:00:12	1-2(1)		1
00:00:13	1-3(7)		7
...			
00:00:18	1-3(1)		1

問題2-3

キャパシティおよび優先度を考慮した上で、より効率よく実行できるワーカーを作成してください。何を指標として、どのように実行スケジュールを工夫したかをレポートに含めてください。

問題3 (自由回答)

※学業に支障の出ない範囲で取り組んでください。

以下からどれか1つを選択して、レポートにまとめてください。

問題3-1

ジョブの優先度が "High", "Low" の二値ではなく、`[0-100]` のような数値に対応したワーカーを作成してください。この場合、`100` が最も優先されるジョブとなります。課題ディレクトリに含まれる以下のデータを使用してください。

- job_list_3-1.zip (MD5: `6cfbd728c042fd8298935941f0b22d5b`)

問題3-2

ワーカーを複数実行できるようにしてください。キャパシティを10に設定した複数のワーカーと、キャパシティを15に設定した1つのワーカーの実行効率を比較してください。

問題3-3

ポイントが偶数の場合は2倍の速さで実行が完了できるとします。例えば、5ポイントの場合は実行に5秒かかりますが、6ポイントの場合は3秒で実行が完了とします。この条件において、効率的にジョブを実行できるワーカーを実装してください。