スポットインスタンスにおける実測に基づく実行条件決定モデルの構築

那須 渚[1,a)]　神野 敦哉[1,b)]　田口 開智[1,c)]　小比賀 亮仁[1,d)]　小板 隆浩[1,e)]

概要：スポットインスタンスは，クラウド上の余剰計算資源を低価格で利用できる一方，稼働中に中断される可能性がある．中断による処理の再実行は，処理完了までの時間を延ばし，計算資源の利用費用も増加させる．既存研究では Spot Placement Score（SPS）を用いた起動先の選択手法が検討されてきた．しかし SPS は起動のしやすさを示す指標であり，本研究の測定では，SPS が同じ値を示す状況でも起動後の中断率が時間帯によって異なっていた．そこで本研究では，実測データに基づく実行条件決定モデルを提案する．まず，所定の時間まで中断されずに稼働する割合を信頼性として定義し，AWS 上で 2 期間計 15 日間の測定を実施した．その結果，信頼性はリージョン・インスタンスタイプ・時間帯によって異なる一方，稼働時間の経過とともに中断されやすくなる傾向は見られなかった．次に，再実行によって生じる時間的損失を「損失時間」と定義し，その値を最小化する実行条件決定モデルを構築した．実測データに基づくシミュレーションにより，実行開始時刻の選択によって損失時間を 33%削減できることを確認した．

キーワード：スポットインスタンス，クラウドコンピューティング，中断，実行条件決定，損失時間

1. 背景

クラウドコンピューティングの普及に伴い，計算資源の利用コストは多くの利用者にとって重要な課題となっている．コストを削減する手段の一つにスポットインスタンスがある [1]．スポットインスタンスは，クラウドプロバイダの余剰計算資源を低価格で提供する料金モデルであり，オンデマンドインスタンスと比較して最大約 90%の割引価格で利用できる．この価格優位性から，機械学習やバッチ処理に加え，CI/CD における自動ビルドやテストのように一時的に計算資源を必要とするワークロードにも利用されている [2]．

1.1 スポットインスタンスの中断

スポットインスタンスは余剰容量に基づいて提供されるため，需要の増加や利用可能な容量の減少が生じると，プロバイダは稼働中のインスタンスを利用者の都合とは無関係に回収する．本稿ではこれを中断と呼ぶ．AWS では中断の約 2 分前に，警告が Amazon EventBridge のイベントおよびインスタンスメタデータとして通知される [3]．ただしこの通知はベストエフォートで提供されるものであり，また 2 分間で可能な対処は限られる．

中断は利用者に次の困難をもたらす．

- **再実行による損失**：チェックポイントを取得していない場合，中断までの処理内容は失われ，ワークロード全体の再実行が必要となる．その結果，処理完了までの時間と利用費用がともに増加する．
- **予測の困難さ**：中断がいつ発生するかを利用者が事前に知ることはできない．
- **条件による変動**：中断のされやすさはリージョン，インスタンスタイプ，時間帯によって異なる．しかし，利用者がこの変動を把握する手段は限られている．

これらの困難は，CI/CD のビルドやテスト，短時間のバッチ処理のような短時間ワークロードにおいて特に問題となる．こうしたワークロードは実行時間こそ短いものの，完了前に中断されればジョブ全体を再実行する必要があり，中断のたびに実行時間と費用が積み重なる．したがって利用者にとって重要なのは，どのリージョン，インスタンスタイプ，実行時刻で実行すれば処理を中断されずに完了できるかという判断である．本稿では，これらの組合せを**実行条件**と呼ぶ．

[1] 同志社大学
[a)] ctwn0144@mail4.doshisha.ac.jp
[b)] ctwp0126@mail4.doshisha.ac.jp
[c)] ctwp0168@mail4.doshisha.ac.jp
[d)] akohiga@mail.doshisha.ac.jp
[e)] tkoita@mail.doshisha.ac.jp

ⓒ2026 Information Processing Society of Japan

1.2 Spot Placement Score

この判断を支援する指標として，AWS は Spot Placement Score（SPS）を提供している [4]．SPS は，指定したインスタンスタイプ，リージョン，必要容量の条件下でスポットリクエストが成立する見込みを 1 から 10 のスコアで示す指標である．スコアが高いほど，要求した容量を確保できる見込みが高い．SPS はリージョン単位またはアベイラビリティゾーン単位で取得できる．既存研究では，SPS を用いて起動先やマイグレーション先を選択する手法が検討されてきた [5]．

しかし SPS が対象とするのはリクエスト時点での容量の確保，すなわち起動のしやすさであり，起動したインスタンスが必要な処理時間だけ中断されずに稼働し続けられるかは対象としていない．短時間ワークロードの利用者にとって，起動に成功することと処理が完了することは別の問題である．本研究ではこの違いを，**起動のしやすさ**と**起動後の継続性**の区別として扱う．

1.3 本研究の目的と貢献

以上を踏まえ，本研究では，プロバイダ固有の指標に依存せず，自ら測定した起動・中断の記録のみに基づいて実行条件を決定するモデルを提案する．本稿の貢献は次の 3 点である．

- 起動の成否と起動後の中断を分離して記録する測定フレームワークを実装し，所定の期限まで中断されずに稼働する割合を期限条件付き完了可能性として定義した．
- AWS の 4 リージョン・3 インスタンスタイプを対象とした 2 期間計 15 日間の測定（起動リクエスト 4,320 件）により，期限 60 分の完了可能性が実行条件によって 0.73 から 1.00 まで変動すること，および稼働時間の経過とともに中断ハザード率が増加する傾向は見られないことを示した．
- 再実行による時間的損失を損失時間として定式化し，入力がすべて実測値である実行条件決定モデルを構築した．測定データを用いたシミュレーションにより，実行開始時刻の選択によって損失時間が 33%削減されることを示した．

以降，2 章で関連研究を整理し，3 章で SPS と起動後の継続性の関係を測定データにより確認する．4 章で信頼性の指標と測定方法を，5 章で測定結果を示す．6 章で実行条件決定モデルを構築し，7 章で評価する．8 章で考察と本研究の限界を述べ，9 章でまとめる．

2. 関連研究

2.1 スポットインスタンスの可用性向上

Lin ら [6] はスポットインスタンスの可用性を向上させる手法を調査し，チェックポイント，マイグレーション，レプリケーション，スケジューリングといったアプローチに整理している．本研究が扱う実行条件の選択はスケジューリングに相当する．

2.2 プロバイダの指標に基づく選択

Katayama ら [5] は，中断発生時のマイグレーション先として SPS がより高いリージョンまたはアベイラビリティゾーンを選択する手法を提案し，再起動の成功率を高められることを示した．また Garg ら [7] は，スポット価格の変動を考慮した入札戦略によりクラウド費用を削減する手法を検討している．これらの手法は起動時点の可用性や費用の観点からは有効であるが，起動したインスタンスがワークロードの完了まで稼働し続けられるかは，これらの指標からは直接判断できない．

2.3 中断挙動の観測

Kim ら [8] は，SPS が時間帯や曜日に応じて周期的に変動すること，および SPS とスポットインスタンスの生存時間との間に一定の関係があることを示している．Lee ら [9] は，SPS，スポット価格，中断率などを継続的に収集するアーカイブサービス SpotLake を提案し，GPU インスタンスや大規模なインスタンスタイプで可用性が低い傾向を報告している．これらの研究は中断挙動の統計的性質を明らかにしているが，観測された変動は主に記述・分析の対象として扱われており，実行条件を決定するための入力として用いる枠組みは十分に検討されていない．

本研究は，プロバイダ固有の指標ではなく自ら測定した中断の記録を用い，それを実行条件の決定に直接用いる点でこれらの研究と異なる．

3. SPS と起動後の継続性

SPS は起動のしやすさを示す指標である．一方，Kim ら [8] は SPS と生存時間の間に一定の関係があることを報告している．そこで本章では，SPS を用いて中断されにくい実行条件を選べるかを，本研究の測定データにより確認する．

3.1 方法

4 章で述べる測定のうち，SPS を記録した 2026 年 10 月 10 日から 18 日までの 8 日間（192 時間）のデータを用いる．対象は 4 リージョン・3 インスタンスタイプである．起動リクエストは 2,304 件，起動成功は 2,114 件であり，このうち 124 件が起動から 60 分以内に中断された．SPS には，リクエスト時に記録した値を用いる．以下では，起動したインスタンスが期限 d まで中断されずに稼働した割合を $F_{\mathrm{comp}}(d)$ と書く（4 章で定義する）．

観測された SPS は，インスタンスタイプごとにほぼ一定の値をとった．各タイプの起動リクエストは 768 件であ

ⓒ2026 Information Processing Society of Japan

表 1 SPS が通常値 (3) のときの時間帯別の $F_{\mathrm{comp}}(60)$ (c6a.large, 4 リージョン合計)

時間帯	$F_{\mathrm{comp}}(60)$	95%信頼区間	n
Night	0.99	0.96–1.00	190
Morning	0.92	0.87–0.95	162
Afternoon	0.92	0.87–0.95	152
Evening	0.97	0.94–0.99	184

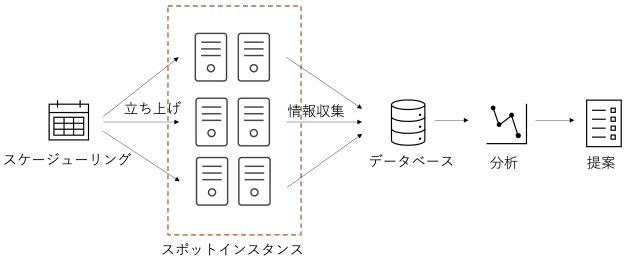

図 1 測定フレームワークの概要

り，SPS は c6a.large ではその 90%で 3，g4dn.xlarge では 57%で 9 であった．いずれのタイプでも，それ以外の大部分は 1 であった．そこで各タイプで最も多く観測された値を通常値とし，SPS が通常値を下回った場合を **SPS 低下**とする．t3.micro は SPS がほぼ常に 3 であり，中断も 1 件のみであったため，分析から除く．

3.2 SPS の低下と中断

SPS 低下時の $F_{\mathrm{comp}}(60)$ は，c6a.large で 0.72（95%信頼区間 0.60–0.81，$n = 67$），g4dn.xlarge で 0.69（0.61–0.76，$n = 158$）であり，SPS が通常値のとき（いずれも 0.95）より低かった．リージョンとインスタンスタイプで層別した Mantel-Haenszel 推定によると，SPS 低下時に 60 分以内に中断されるオッズは通常時の約 7.0 倍（95%信頼区間 4.5–10.8）であった．したがって，SPS は起動後の中断とも関連しており，この結果は Kim ら [8] の報告と整合する．

ただし，SPS の低下は個々の起動の中断を予測するものではない．SPS 低下時に起動したインスタンスの 70%は 60 分間中断されず，一方で分析対象の中断 123 件のうち 55 件（45%）は SPS が通常値のときに発生した．また，起動に成功したインスタンスの大部分（c6a.large で 91%，g4dn.xlarge で 73%）は SPS が通常値のときに起動しており，このとき SPS は実行条件の間に差をつけない．

3.3 同じ SPS のもとでの継続性の差

SPS では区別できない継続性の差があるかを調べるため，SPS が通常値のときの起動に限って $F_{\mathrm{comp}}(60)$ を時間帯別に求めた．**表 1** に示すように，c6a.large では SPS がすべて 3 であるにもかかわらず，$F_{\mathrm{comp}}(60)$ は Night の 0.99 に対し Morning と Afternoon では 0.92 であった．中断率に換算すると Night の 1.1%に対し Morning は 8.0%であり，95%信頼区間は重ならない（時間帯間の差の χ^2 検定，$p = 0.002$）．なお，g4dn.xlarge では同様の差は有意ではなかった（$p = 0.43$）．また，両タイプを合わせ，SPS 低下，リージョン，インスタンスタイプを説明変数とするロジスティック回帰を当てはめた．これに時間帯を加えると，当てはまりは有意に改善した（尤度比検定，$p < 0.001$）．すなわち，時間帯による中断されやすさの差の一部は SPS に現れない．

3.4 まとめ

以上より，SPS は中断と関連するものの，SPS が同じ値を示す状況での継続性の差を区別することはできない．時間帯による差には SPS に現れない成分が含まれるため，SPS のみに基づく選択ではこの差を利用できない．また，SPS の通常値はインスタンスタイプによって異なり（c6a.large では 3，g4dn.xlarge では 9），両者の $F_{\mathrm{comp}}(60)$ は同程度であったことから，SPS の値はタイプ間の比較にも用いることができない．中断されにくい実行条件を選ぶには，実行条件ごとの完了可能性を実測に基づいて求める必要がある．

4. 期限条件付き完了可能性の測定

4.1 指標の定義

本研究では，起動したインスタンスが所定の期限まで中断されずに稼働する割合を指標として用いる．測定では実際のアプリケーションは実行せず，期限 d まで中断されずに稼働し続けたことをもって，処理時間が d 以下のワークロードが完了可能であったとみなす．

起動リクエスト数を N_{req}，起動に成功したインスタンス数を N_{succ}，起動に成功したインスタンスのうち期限 d より前に中断された数を $N_{\mathrm{int}}(d)$ とする．起動成功率 R_{launch} と期限前中断率 $R_{\mathrm{int}}(d)$ を次のように定義する．

$$R_{\mathrm{launch}} = \frac{N_{\mathrm{succ}}}{N_{\mathrm{req}}}, \qquad R_{\mathrm{int}}(d) = \frac{N_{\mathrm{int}}(d)}{N_{\mathrm{succ}}} \tag{1}$$

そのうえで，**期限条件付き完了可能性** $F_{\mathrm{comp}}(d)$ を次式で定義する．

$$F_{\mathrm{comp}}(d) = 1 - R_{\mathrm{int}}(d) \tag{2}$$

R_{launch} と $F_{\mathrm{comp}}(d)$ を分けて定義することで，起動時点の可用性と起動後の継続性を区別して扱える．SPS が対象とするのは前者であり，$F_{\mathrm{comp}}(d)$ が対象とするのは後者である．両者は排他的ではなく，信頼性の異なる側面を捉える相補的な指標である．

以降，$F_{\mathrm{comp}}(d)$ の値を完了確率と呼ぶ．

4.2 測定フレームワーク

図 1 に測定フレームワークの概要を示す．本フレームワークは Terraform により各リージョンに同一の構成で構築し，AWS Lambda，Amazon EventBridge，Amazon

表 2 対象インスタンスタイプ (価格は観測期間中の 4 リージョン平均スポット価格)

タイプ	分類	価格（USD/h）
c6a.large	計算最適化	0.039
g4dn.xlarge	GPU（T4）搭載	0.307
t3.micro	バースト可能	0.005

DynamoDB を用いて実装した．

EventBridge のスケジュールにより起動用の Lambda 関数を毎時 0 分に実行し，対象の各インスタンスタイプについてスポットインスタンスを 1 台ずつ要求する．容量不足などアベイラビリティゾーン単位のエラーでリクエストが失敗した場合は，同一リージョン内の別のゾーンで再試行し，いずれのゾーンでも成立しなかった場合を起動失敗とする．リクエスト時には対象ゾーンのスポット価格を記録し，10 月の測定では SPS もあわせて記録した．

起動したインスタンスは 1 時間の観測期間にわたって稼働し，期間が終了するとインスタンス自身が停止して終了する．中断の検知には二つの経路を用いる．一つはインスタンス内部で 5 秒ごとにインスタンスメタデータを確認する経路であり，もう一つは EventBridge が通知する中断警告イベントを Lambda 関数で受け取る経路である．いずれかの経路で中断警告を検知したインスタンスを中断として記録し，起動からの経過時間をあわせて記録する．また，インスタンスの状態遷移イベントから起動時刻，running 状態への遷移時刻，配置されたゾーンを記録する．これらの記録はすべて DynamoDB に集約する．

4.3 実験設定

対象のインスタンスタイプは，リソース特性の異なる c6a.large，g4dn.xlarge，t3.micro の 3 種類とした [10], [11]（**表 2**）．対象リージョンは，N. Virginia（us-east-1），Ireland（eu-west-1），Tokyo（ap-northeast-1），Singapore（ap-southeast-1）の 4 つである．各リージョンで各インスタンスタイプを 1 時間に 1 台起動し，2026 年 5 月 19 日から 25 日までの 7 日間と，10 月 10 日から 18 日までの 8 日間（192 時間）の計 15 日間測定した．最大スポット価格は指定していない．

時間帯の影響を分析するため，各リージョンのローカル時刻に基づいて 1 日を Night（0–6 時），Morning（6–12 時），Afternoon（12–18 時），Evening（18–24 時）の 4 区間に分割した．リージョン，インスタンスタイプ，時間帯の組合せは 48 通りであり，各組合せにつき 90 件（6 時間 ×15 日）の起動リクエストを送信した．

5. 測定結果

5.1 全体の傾向

測定期間中に $N_{\mathrm{req}} = 4{,}320$ 件の起動リクエストを送信し，このうち $N_{\mathrm{succ}} = 4{,}064$ 件が起動に成功した（$R_{\mathrm{launch}} = 94.1\%$）．起動に成功したインスタンスのうち 231 件（5.7%）が 1 時間の観測期間の終了前に中断された．全データをまとめた $F_{\mathrm{comp}}(d)$ は，$d = 5$，10，30，60 分に対してそれぞれ 0.994，0.988，0.967，0.943 であった．また，起動リクエストからインスタンスが running 状態となるまでの起動オーバーヘッドは，平均 4.42 秒（中央値 4 秒，最大 31 秒）であった．

表 3 N. Virginia，c6a.large における時間帯別の $F_{\mathrm{comp}}(60)$

時間帯	$F_{\mathrm{comp}}(60)$	95%信頼区間	n
Night	0.97	0.91–0.99	90
Morning	0.78	0.68–0.86	82
Afternoon	0.81	0.72–0.88	81
Evening	0.91	0.83–0.95	90

5.2 実行条件による変動

図 2 に，リージョン，インスタンスタイプ，時間帯ごとの $F_{\mathrm{comp}}(60)$ を示す．t3.micro はすべての条件で $F_{\mathrm{comp}}(60) \geq 0.96$ であり，中断の影響を受けにくかった．一方，c6a.large と g4dn.xlarge では条件による変動が大きく，$F_{\mathrm{comp}}(60)$ は 0.73 から 1.00 の範囲で変化した．

時間帯による変動が最も大きかったのは Ireland の c6a.large（0.77–1.00）であり，N. Virginia の c6a.large（0.78–0.97）がこれに次いだ．ただし Ireland の c6a.large では Night に中断が一件も観測されていないため，以降では，すべての時間帯で中断が観測された N. Virginia の c6a.large を例に示す（**表 3**）．Night と Morning の差は約 0.19 であり，Wilson の 95%信頼区間 [12] は重ならない．したがって，この差が標本変動のみによって生じた可能性は低い．

また，$F_{\mathrm{comp}}(60)$ が最も低くなる時間帯はリージョンによって異なっていた．N. Virginia と Ireland の c6a.large では Morning が最も低く，Singapore の g4dn.xlarge では Afternoon が最も低かった．この傾向は，中断が各地域のクラウド需要の日内変動に対応しているという解釈と整合する．

5.3 中断ハザード率

中断の発生率が稼働時間の経過とともに変化するかを調べた．経過時間を区間に分け，各区間の中断件数を，その区間の開始時点で稼働していたインスタンス数で割ってハザード率とした（**図 3**）．ハザード率は 0.057–0.122%/分の範囲で変動したが，経過時間に対して単調に増加する傾向は見られなかった．すなわち，長く稼働したインスタンスほど中断されやすくなるという性質は観測されなかった．ただし，60 分以内の中断 231 件という標本数ではハザード率が一定であることを統計的に強く主張することはできない．本稿では，単調増加を示す証拠は得られなかったという主張にとどめる．

ⓒ2026 Information Processing Society of Japan

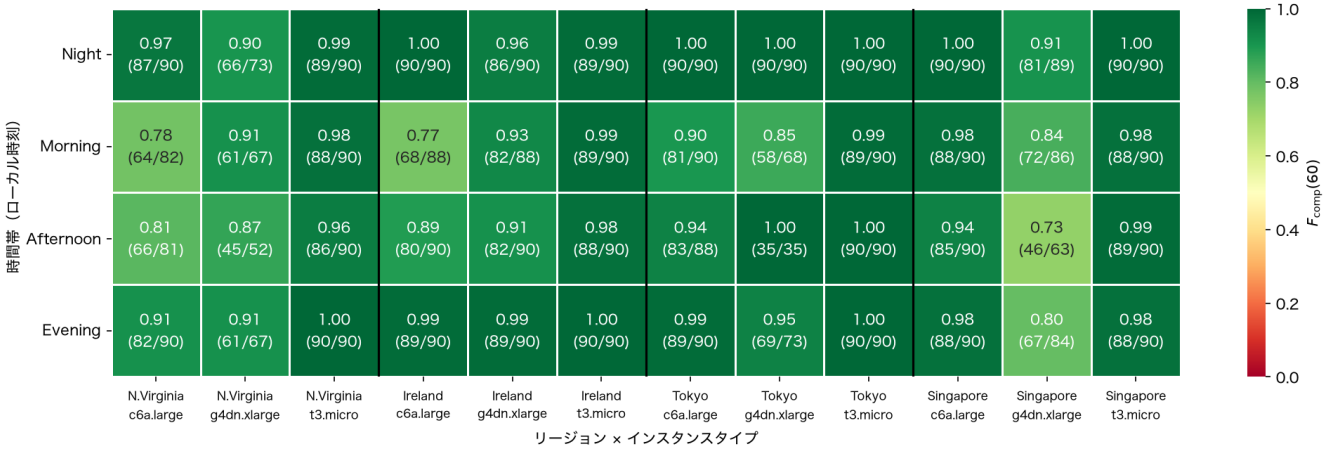

図 2 リージョン，インスタンスタイプ，時間帯別の $F_{\mathrm{comp}}(60)$．括弧内は (60 分間中断されなかった数/起動成功数) を示す．

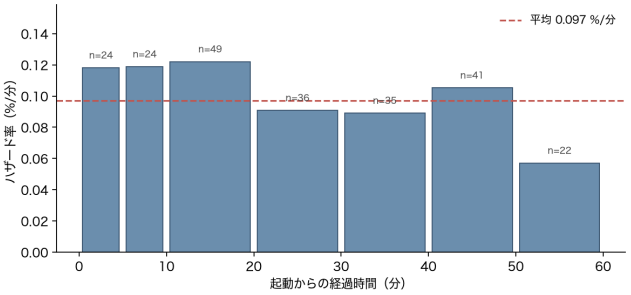

図 3 経過時間別の中断ハザード率．破線は平均値，n は各区間の中断件数を示す．

5.2 節の結果と合わせると，中断のされやすさは稼働時間の長さよりも実行条件に左右されると考える．

6. 実行条件決定モデル

5 章の結果は，中断されにくい実行条件を選ぶことで再実行による損失を減らせることを示唆している．本章では，この損失を損失時間として定式化し，実行条件決定モデルを構築する．

6.1 損失時間

中断されたジョブは再実行されるため，利用者が要する総時間は本来の処理時間を上回る．本稿ではこの超過分を**損失時間**と呼び，モデルの最小化対象とする．損失時間は，再実行回数と，1 回あたりの中断までに費やした時間の積として表される．前者は実行条件の中断されにくさによって決まり，後者は中断が発生するまでの経過時間によって決まる．

ジョブは時刻 t_0 に到着し，処理時間 d を持つとする．また開始時刻を最大 Δ だけ遅らせてよいとし，Δ を許容遅延と呼ぶ．決定変数は，リージョン r，インスタンスタイプ τ，実行開始時刻 t である．時刻 t におけるリージョン r のローカル時間帯を $\pi = \pi(r, t)$ と書く．

実行条件 (r, τ, π) のもとで処理が完了する確率 p は，式 (2) の期限条件付き完了可能性そのものである．

$$p(r, \tau, \pi, d) = F_{\mathrm{comp}}^{(r,\tau,\pi)}(d) \tag{3}$$

各試行が独立であると仮定すると，成功するまでの試行回数は幾何分布に従い，その期待値は $1/p$ である．したがって，再実行回数の期待値は $1/p - 1$ となる．また，中断された試行で費やされた時間の期待値を

$$\bar{u}(r, \tau, \pi, d) = \mathrm{E}\left[U \mid U < d\right] \tag{4}$$

とする．ここで U は起動から中断までの稼働時間である．以上より，損失時間 L を次式で定義する．

$$L(r, \tau, t) = \left(\frac{1}{p(r, \tau, \pi, d)} - 1\right) \bar{u}(r, \tau, \pi, d) + s \tag{5}$$

ここで s は起動オーバーヘッドである．p，$\bar{u}$，s はいずれも 5 章の測定データから直接算出でき，プロバイダ固有の指標を必要としない．

6.2 設計上の判断

最小化対象．最小化対象として利用料金も考えられるが，本研究では損失時間を用いる．料金を目的関数とすると，リージョン間の価格差が中断特性の差を覆い隠すためである．たとえば c6a.large の観測スポット価格は，N. Virginia の 0.0301 USD/h に対し Tokyo では 0.0445 USD/h と 48%高い．一方，中断による損失時間は，N. Virginia の c6a.large で平均 4.06 分（**表 5**）であり，1 時間のジョブに対して 6.8%にとどまる．このため料金を最小化すると，中断特性によらず常に最安のリージョンが選ばれる．

推定誤差への対処．完了確率 p は，条件あたり高々 90 件の標本から推定した値である．最適化は最も良く見える条件を選ぶため，点推定値をそのまま用いると偶然良好な値を示した条件を選びやすい．そこで条件の選択には，p

表 4 到着時刻ごとの決定例 (N. Virginia，c6a.large，$d = 60$ 分，$\Delta = 6$ 時間)．到着時刻は UTC，L の単位は分，遅延の単位は時間．

到着	時間帯	L（即時）	遅延	実行時間帯	L（決定後）
0 時	Evening	3.21	4	Night	1.31
4 時	Night	1.31	0	Night	1.31
10 時	Morning	6.65	6	Afternoon	5.07
16 時	Afternoon	5.07	6	Evening	3.21
22 時	Evening	3.21	6	Night	1.31
24 時刻の平均		4.06	2.6		2.73

の代わりに Wilson の 95%下側信頼限界 $\tilde{p}$[12] を用いる．$\tilde{p}$ は標本数が少ないほど点推定値から大きく下がるため，標本の少ない条件は選ばれにくくなる．なお，効果の評価には点推定値 p を用いる．

6.3 最適化問題

実行条件の選択は，次の最適化問題として記述される．

$$\begin{aligned} \min_{r,\tau,t} \quad & L(r,\tau,t) \\ \text{s.t.} \quad & t_0 \le t \le t_0 + \Delta \\ & r \in \mathcal{R}, \quad \tau \in \mathcal{T} \\ & p(r,\tau,\pi(r,t),d) \ge p_{\min} \end{aligned} \tag{6}$$

$\mathcal{R}$，$\mathcal{T}$ は利用可能なリージョンとインスタンスタイプの集合，$p_{\min}$ は利用者が求める完了確率の下限である．$\Delta = 0$ のとき $t = t_0$ となり，選択できるのはリージョンとインスタンスタイプのみとなる．実行開始時刻を 1 時間単位で選ぶ場合，候補は高々 $4 \times 3 \times 24 = 288$ 通りであり，全列挙により厳密に解ける．

6.4 決定手続き

式（6）は次の 2 段階で解く．第 1 段階では，測定データから実行条件ごとの p，$\tilde{p}$，$\bar{u}$ を算出し，表として保持する．条件数は 48 であり，表は小さい．第 2 段階では，ジョブの到着時に許容遅延の範囲で候補を列挙し，損失時間が最小となる条件を選ぶ．

表 4 に，リージョンとインスタンスタイプを固定し実行開始時刻のみを選択する場合の決定例を示す．この例から次の 3 点が確認できる．第一に，常に遅延させるわけではない．24 時刻のうち 6 時刻では遅延なしで実行される．第二に，必ずしも最良の時間帯を選ぶわけではない．10 時に到着した場合，最良である Night には許容遅延 6 時間では到達できないため，到達可能な範囲で最良の Afternoon を選ぶ．第三に，平均遅延は 2.6 時間であり，許容遅延の上限まで待つ必要は必ずしもない．

7. 評価

7.1 評価方法

本章の評価は，5 章の測定データを用いたシミュレーションである．各戦略が選んだ実行条件に対し，測定データから求めた p と $\bar{u}$ を用いて式（5）の損失時間を算出する．実際にワークロードを実行して比較したものではない．

表 5 許容遅延 Δ に対する損失時間 (N. Virginia, c6a.large, $d = 60$ 分)

Δ	0 時間	3 時間	6 時間	12 時間
B0 の損失時間 [分]	4.06（Δ によらず一定）			
B1 の損失時間 [分]	4.06	3.39	2.73	1.79
削減率	—	16%	33%	56%
B1 の平均完了確率	0.868	0.892	0.915	0.953

表 6 戦略ごとの損失時間 ($d = 60$ 分，$\Delta = 6$ 時間)

戦略	選択対象	L [分]	削減率
B0	なし	4.06	—
B1	実行開始時刻	2.73	33%
B2	リージョン	0.37	91%
B3	リージョン，実行開始時刻	0.12	97%

式（6）の探索範囲を制限することで，次の 4 つの戦略を定義する．B0 は到着時刻にそのまま実行する戦略であり，ベースラインとする．B1 は実行開始時刻のみを選択する戦略，B2 は待機せずにリージョンのみを選択する戦略，B3 はリージョンと実行開始時刻の両方を選択する戦略である．いずれの戦略でもインスタンスタイプは c6a.large とし，B0 と B1 ではリージョンを N. Virginia に固定する．処理時間は $d = 60$ 分とし，ジョブは毎時 0 分（UTC）に到着するものとして，24 通りの到着時刻にわたる平均を求める．

7.2 実行開始時刻の選択

表 5 に，許容遅延 Δ を変化させたときの B1 の損失時間を示す．$\Delta = 6$ 時間のとき，実行開始時刻の選択により損失時間は 4.06 分から 2.73 分へ 33%削減され，平均完了確率は 0.868 から 0.915 へ改善した．許容遅延が大きいほど到達できる時間帯が増えるため，削減率は $\Delta = 12$ 時間で 56%まで増加する．

7.3 実行場所の選択

表 6 に，リージョンまで選択範囲を広げた場合の結果を示す．B2 と B3 は B1 を大きく上回る削減を示しているが，この値はそのまま受け取るべきではない．B2 と B3 が主に選択するのは，Tokyo や Ireland の Night（いずれも $n = 90$）のように，測定期間中に中断が一件も観測されなかった条件である．こうした条件では完了確率の推定値が 1.00 となり，損失時間はほぼ 0 と評価される．

48 条件のうち 11 条件（23%）では，中断が一件も観測されていない．条件あたり 90 件程度の標本では，中断率が低い条件ほど偶然 0 件となりやすい．たとえば中断率 1%の条件でも，90 件中 0 件となる確率は約 40%である．したがって完了確率 1.00 は，中断しないことではなく，中断

ⓒ2026 Information Processing Society of Japan

が観測されなかったことを意味する．Wilson の下側信頼限界は標本数の少ない条件を回避するが，標本数が同程度で中断 0 件の条件どうしを区別することはできない．

以上より，本稿では B1 を主たる結果とする．B1 は同一リージョン内での時間帯の比較であり，比較するすべての時間帯で中断が実際に観測されている（表 3）．B2 と B3 についてはさらなる削減の余地を示唆するにとどめ，効果の確定には標本数の増加が必要である．

8. 考察

8.1 実行条件の選択が有効な理由

中断ハザード率には，経過時間とともに増加する傾向が見られなかった（5.3 節）．一方，期限 60 分の完了可能性は実行条件によって 0.73 から 1.00 まで異なった（5.2 節）．これらは，少なくとも 60 分以内のジョブでは，中断のリスクが稼働時間の長さよりも実行条件に左右されることを示唆する．実行開始時刻の選択が損失時間を削減できたのは，この性質によると考える．

8.2 本研究の限界

本研究には次の限界がある．第一に，測定期間は 2 期間計 15 日間であり，条件ごとの起動成功数は 35–90 件である．時間帯ごとの傾向は期間によって変化しうるため，測定結果が有効な期間は明らかでない．第二に，中断されずに稼働したことを処理完了の代理指標としており，実際のアプリケーションは実行していない．第三に，各条件で 1 時間に 1 台のみを起動しているため，同時に多数のインスタンスを起動した場合の挙動は捉えていない．第四に，最大スポット価格を指定しない設定での挙動のみを扱っている．第五に，7 章の評価は，条件の選択と効果の測定に同一の測定データを用いたシミュレーションであり，試行の独立性を仮定している．したがって 33%は測定データ上での値であり，測定期間外に適用した場合の効果はこれより小さくなりうる．

今後は，提案モデルを実環境で検証する実証実験を行う．到着時刻にそのまま実行する対照群と，モデルが決定した時刻に実行する処理群を同一期間に並行して実行し，両群の完了率と損失時間を比較する．完了確率の差（4.7 ポイント）を有意水準 5%，検出力 80%で検出するには両群合計で約 1,400 件の試行が必要であり，1 日 48 件の実行で約 30 日を要する．ただし，期間を越えた効果はこれより小さい可能性があり，その場合に必要な試行数はさらに増える．

9. おわりに

本稿では，スポットインスタンス上の短時間ワークロードを対象に，実測データに基づいて実行条件を決定するモデルを提案した．まず，SPS を記録した測定データの分析により，SPS は中断と関連するものの，SPS が同じ値を示す状況でも中断率が時間帯によって異なり，SPS だけでは中断されにくい実行条件を選べないことを確認した．次に，所定の期限まで中断されずに稼働する割合を期限条件付き完了可能性として定義し，AWS の 4 リージョン・3 インスタンスタイプを対象に 2 期間計 15 日間の測定を行った．その結果，期限 60 分の完了可能性は実行条件によって 0.73 から 1.00 まで変動した．一方，中断ハザード率には稼働時間の経過とともに増加する傾向は見られなかった．さらに，再実行による時間的損失を損失時間として定式化し，入力がすべて実測値である実行条件決定モデルを構築した．シミュレーションによる評価の結果，許容遅延 6 時間のもとで実行開始時刻を選択することにより損失時間を 33%削減できることを示した．

今後の課題として，実環境での実証実験，測定期間の延長による長期的な傾向の把握と測定結果の有効期間の検討，実行場所の選択による効果の確定，および SPS と実測に基づく完了可能性を組み合わせた実行条件の選択が挙げられる．これらを通じて，提供側の指標だけに頼らず，利用者自身の測定に基づいてスポットインスタンスを活用する枠組みの確立を目指す．

参考文献

[1] Amazon Web Services: Amazon EC2 Spot Instances(オンライン)，入手先 ⟨https://aws.amazon.com/ec2/spot/⟩（参照 2026-10-23）.

[2] Ramesh, D.G.A. and Lennon, R.G.: Evaluating the Impact of Reserved Instances vs. Spot Instances in DevOps Cloud Workflows, *Proc. 2025 35th Irish Signals and Systems Conference (ISSC)*, pp.1–7 (2025).

[3] Amazon Web Services: Spot Instance Interruption Notices, Amazon EC2 User Guide(オンライン)，入手先 ⟨https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-instance-termination-notices.html⟩（参照 2026-10-23）.

[4] Amazon Web Services: Spot Placement Score, Amazon EC2 User Guide(オンライン)，入手先 ⟨https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-placement-score.html⟩（参照 2026-10-23）.

[5] Katayama, D., Kasai, K. and Koita, T.: Migration Destination Selection Algorithm for Spot Instances Using SPS, *Proc. 2022 IEEE International Conference on Big Data (Big Data)*, pp.6690–6692 (2022).

[6] Lin, L., Pan, L. and Liu, S.: Methods for Improving the Availability of Spot Instances: A Survey, *Computers in Industry*, Vol.141, p.103718 (2022).

[7] Garg, D., Kaata, S.K., Bhumireddy, D.R. and Bolla, B.K.: Reducing Cloud Cost Through Optimized Bidding for AWS Spot Instances, *Proc. 2025 3rd International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT)*, pp.1–7 (2025).

[8] Kim, K. and Lee, K.: Making Cloud Spot Instance Interruption Events Visible, *Proc. ACM Web Conference 2024 (WWW '24)*, pp.2998–3009 (2024).

[9] Lee, S., Hwang, J. and Lee, K.: SpotLake: Diverse Spot Instance Dataset Archive Service, *Proc. 2022 IEEE International Symposium on Workload Characterization (IISWC)*, pp.242–255 (2022).

ⓒ2026 Information Processing Society of Japan

[10] Pham, T.-P., Ristov, S. and Fahringer, T.: Performance and Behavior Characterization of Amazon EC2 Spot Instances, *Proc. 2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp.73–81 (2018).
[11] Amazon Web Services: Amazon EC2 Instance Types, Amazon EC2 User Guide(オンライン), 入手先 ⟨https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html⟩（参照 2026-10-23）.
[12] Wilson, E.B.: Probable Inference, the Law of Succession, and Statistical Inference, *Journal of the American Statistical Association*, Vol.22, No.158, pp.209–212 (1927).

ⓒ2026 Information Processing Society of Japan