日本データベース学会和文論文誌
Vol. 24-J, Article No. 12, 2026 年 3 月

全固体電池研究のための Low-Cost 統合実験 Database System の開発

柏山 正守[1]　中村 武雄[2]　神谷 倫代[3]
押野谷 恵美[4]　松井 直喜[5]　堀 智[6]
鈴木 耕太[7]　菅野 了次[8]

全固体電池研究の実験過程では，膨大な量の高次元・非正規構造の計測・解析データが生まれる．これら実験データを収納し，マテリアルズ・インフォマティクスへ適用する統合実験データベース基盤を開発した．当該データベースは，クラス最大規模の全固体電池材料データベースであり，45,000 サンプル・3,100 万項目を収納する．本開発では，実験フローと対をなすデータ収納方式と計算量理論に基づく性能推定アルゴリズムを提案し，実験により有効性を確認した．本開発におけるデータ収納方式と性能推定アルゴリズムを応用することにより，研究・産業用途向けの高性能データベースを低コストで構築することが可能となる．

1. 序論

全固体電池（All-Solid-State Battery）の研究開発に実装する，統合実験データベース基盤（Integrated Experimental Database System : IEDB System）の方式提案と有効性検証を実施した．本開発の成果を類似な工程を持つ他の製造・研究のデジタル設備へ適用することにより，容易な Database システムの実現と運用，および効率的な利活用が可能となる．全固体電池の実験では，さまざまな固体から電解質と電極を合成し，それらを組み合わせてバッテリー（Battery）を創り出す．東京科学大学総合研究院全固体電池研究センター（本研究センター）では，４つの実験グループ「硫化物系（Sulfides），酸化物系（Oxides），フッ化物系（Fluorides）およびフッ化物のコンビナトリアル（Fluoride Combinatorial）系」の研究者が多種多様な合成と計測・解析を繰り返す．一連の実験過程においては，実験装置・計測装置が相互に連携するため，膨大な量の高次元・非正規構造の計測・解析データが生まれる．現在までに蓄積されたデータ量は，サンプル数：約 45,000 件，これらサンプルの総項目数：約 3,100 万個であり，これらのデータを一元的に管理し，循環と相互利用を円滑・効率良く推進するため IEDB System の開発が必要であった．ところで，本研究センターのデータ蓄積量は，全固体電池の実験 Database としては最上級の規模であることを付け加える．一般的に Database の構築のためには，実験プロセス・フローに従った共通テンプレートを策定し記録する必要がある．しかし，この共通テンプレートへの変換作業は，実験グループ・ユニークなオリジナル記録が持つ追跡性（Traceability）や遡行性（Retrograde）が失われるリスクが伴う．そこで，IEDB System では，高次元・非正規構造データと親和性の高い NoSQL の MongoDB DBMS を採用し，グループ・ユニークな実験記録シートとローカル Database を包括実装するスケーラビリティの高い論理・実装方式を発想した．当該方式は，python script で作成した共通 API（Application Programing Interface）を介して，一元的にグループ個別と全体のデータトランザクション（Transaction）の実行が可能である．さらに，当該方式は実験グループや実験バリエーションの追加が柔軟におこなえ，高い拡張性と変更性を持つ．この共通 API により Web からの閲覧・検索と，データ駆動型材料探索（Materials Informatics : MI）のためのデータセット（Dataset）抽出にも対応可能なデータハブ基盤構造となっている．Transaction 性能に関しては，IEDB System の最多用データ操作である検索性能に着目し，その計算量の理論（Complexity Order algorithm）と CPU（RISC プロセッサ）のマイクロアーキテクチャ（Micro Architecture）性能，およびメモリサブシステム（Memory Subsystem）性能の融合により，実測値に近似する性能推定アルゴリズムを導出し適用した．最終的に IEDB System 利用において，待ち時間の少ないレスポンス性能を実現した．

さらに，ハードウェア・プラットフォームに関しては，RISC プロセッサ SoC（System on a Chip）が持つポテンシャルに解釈を与え，今後の Database 開発におけるデータ量と性能のシンプルなサイジング手法を提案した．本論文を応用すれば，比較的容易に Low-Cost Database の構築が可能となる．

2. 課題の説明

2.1 実験 Database の重要性

近年，全固体電池が次世代の蓄電池として大きな期待を集めている．全固体電池の特長は，高いエネルギー密度，安定した化学反応，高いイオン伝導性などがあり，用途はスマートデバイスから定置型蓄電に至る．全固体電池研究の成果は，循環型社会実現に向けて，我が国の産業発展に多大な効果があると考えられている．

このような多くの利点を持つ全固体電池研究では，さまざまなマテリアルズ（Materials）を組み合わせた合成実験と，それらの測定・解析が確実な実用化へつなげる重要なプロセスとなっている．また，当該プロセスを効率良く進めていくため，使い勝手の良いデジタル・データ基盤「実験 Database」を整備することは非常に重要であった．

そこで本研究センターでは，10 数年間の膨大な実験データを整理し，データ保管と履歴管理，データアクセスおよび解析が

[1] 正会員 東京科学大学総合研究院全固体電池研究センター 技術支援員 kashiyama.m.e9b3@m.isct.ac.jp
[2] 非会員 同センター 技術支援員 nakamura.t.f0d6@m.isct.ac.jp
[3] 非会員 同センター 研究員 kamiya.m.4692@m.isct.ac.jp
[4] 非会員 同センター 技術支援員 oshinoya.e.177b@m.isct.ac.jp
[5] 非会員 同センター 助教授 matsui.n.ee49@m.isct.ac.jp
[6] 非会員 同センター 特任准教授 hori.s.f341@m.isct.ac.jp
[7] 非会員 同センター 准教授 suzuki.k.f71a@m.isct.ac.jp
[8] 非会員 同センター長 特命教授 kanno.r.ade9@m.isct.ac.jp

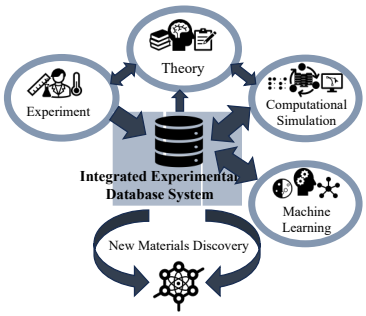


Fig. 1. Integration and Synergy.

一元的に管理できる IEDB System の実現を目指した．

開発では，複数のグループが行った実験条件と測定・解析データの関係をどのように IEDB System として実現し，かつ Transaction 性能を担保できるかが重要な課題であった．

さらに，Fig.1 に示すようにデータ基盤のハブとして，理論（Theory）や計算シミュレーション（Computational Simulation），機械学習（Machine Learning : ML）と連携させ MI による高効率な材料探索を実行する．これら MI 探索に際して，効率良く計算・ML 処理の Dataset を生成し，データ連携を容易化する API を備えることも必要であった．

2.2 統合実験 Database System のデザイン要件

Materials から最終的な電池性能の測定に至る過程では，複数の合成と測定・解析を異なる研究者が進めていく都合上，データの連携・循環や相互利用を効率的におこなえる機能も必要であった．Fig.2 は，前述したグループ毎におこなう実験と測定・解析フローの概要を表現したものである．図は実験・測定・解析の流れを標準的に示しており，実際は Materials や合成手法「合成条件」の相違により複雑な分岐が生じるような実験プロセス・フローとなっている．図示しないがそれぞれのグループが記録するデータはグループ・ユニークなフォーマットを持ち，かつ測定・解析結果のグラフなど，イメージデータに類似するものも含まれた高次元の非構造化データになっており，オリジナルデータをストレートに収納する方式が必要であった．

ところで，高次元・非構造化 Database System 開発に関して，定量的な Transaction 性能推定方式やアーキテクチャ・デザイン要件に関する情報や文献は少なく，IEDB System 開発のデザインを決定していく課程では，先行した Sulfides のグループ・ユニークな Database（MongoDB）の方式をベースとし，性能推定理論を組み立てる必要があった．

3. 既往研究からの発想

3.1 データ構造と収集方式

文献[1，2]で Fujii，Yamazaki らは，プラズマ実験における実験データ収集・解析インフラストラクチャ構築に関する論文を提案しており，我々の実験環境および IEDB System と類似なデータ基盤となっている．特に文献[2]では，異なるデータフォーマットを一貫した共通データ構造へ変換するのではなく複数の分散サーバと Database に分ける手法を採用しつつアクセスの一元化を与える提案がなされている．

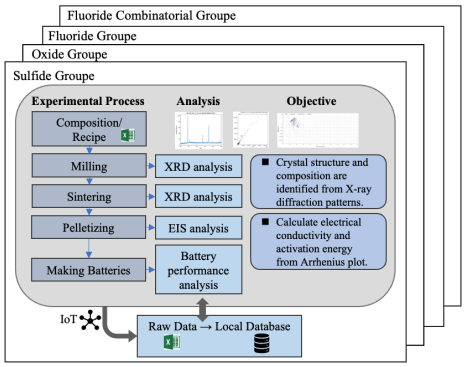


Fig. 2. Experimental and Analysis Process Flow for All-Solid-State Batteries.

そこで，IEDB System では文献[1，2]の考え方を発展させた方式を発想した．本方式は，各実験グループのオリジナル記録データをストレートに個別の Database へ取り込む構成とし，ユーザアクセスにおいては，一元的な仮想 Database 構造となるよう Python Script で高位から Wrapper する方式である．さらに，複数の Database を単一サーバ上に集約する構造とし，効率的に一元管理が可能で，かつユーザアクセスのための API 作成も Python script を用いて自由度が高い開発が可能になる．

ところで，文献[1，2]は有用な提案となっているが，Database の重要なファクターである Transaction 性能のデザイン要件に関する考え方が示されていない．本論文では，この課題を後述の 3 章 3.3 節の発想で解決する．

3.2 NoSQL Database 性能を支配する要素

文献[3]で Cieslewicz らは，Database の Transaction 性能を決定づける Micro Architecture のファクターは，マルチコア（Multi Core）とマルチスレッド（Multi Thread）の並列実行とデータ供給ボトルネックを解消させるキャッシュメモリのサイズ（Cache size），メモリデータバンド幅（Memory bandwidth）性

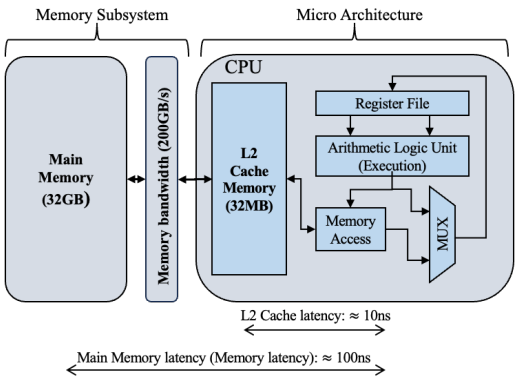


Fig. 3. Computing System Overview, relationship between CPU, L2 Cache, and Main Memory.

能にあると結論付けしている．このことから，MongoDB を動作させる CPU は，①高い CPU 動作周波数，②Multi Core と Multi Thread 動作機構，③大きな Cache size 構成，④高性能な Memory Subsystem を持つことが重要な要素であると考えた．

3.3 計算量と Database 性能

IEDB System の主な利用目的は，検索と解析，および計算・ML への Dataset 抽出であり，これら Transaction の代表的な動作に関する時間コストを計算量オーダー（order of growth）という理論を用いて推定することとした．

文献[4]で Toda は，計算量理論モデルの定式化において，計算機構造の相互関係を調べる必要性があり，検索は計算量理論の判定問題に置き換えが可能であると説明している．この判定問題は，入力データ集合から出力結果の集合への関数 *f(n)*となり，逐次型モデルとして定式化される．この関数 *f(n)*は，計算機構造(Fig.3)からアプリケーションの演算 CPU クロック数である時間計算量（time complexity）と動作するメモリ量である空間計算量（space complexity）のコストを減少させることで総計算量は小さくなると理解できる．計算機構造として考えた場合，Multi Core と Multi Thread の数を増やし，Cache size と Memory bandwidth を拡大し，メインメモリのレイテンシー（Memory latency）を縮退させることが全体計算量低減における計算量感度が高いといえる．これらのことは，CPU の Micro Architecture と Memory Subsystem を高度化するということである．

ところで，一般に２分探索木の計算量は，Eq.(1)で表現される．ここで Nは全体の検索項目数であり，tは処理手数である．

$$\frac{N}{2^t} = 1$$

$$t = log_{10} N \,/\, log_{10} 2 \quad \cdots\cdots (1)$$

しかし，IEDB System のような Database での検索は，比較演算が主であり比較処理が持つ回数が全体計算時間を支配する主要因となることから，その計算量は，Eq.(2)へ漸近でき Eq.(3)が成立すると考えた．

$$Complexity\colon \mathrm{O}\big(nlog(n)\big) comparisons \quad \cdots\cdots (2)$$

$$f(n) \in \Theta\big(nlog(n)\big) \quad \cdots\cdots (3)$$

さらに，Eq.(3)に IEDB System の計算機構造を考慮した前述の計算量オーダーを適用し，平均計算量(Average amount of calculation)のアルゴリズム式として Eq.(4)を発想した．

$$f(n)' = \mathrm{O}\big(nlog(n)\big) \times time\ complexity + space\ compexity \quad \cdots\cdots (4)$$

ここで，*f(n)'*は処理手数に要する時間コストである．さらに，space complexity は，前述の通り，Memory Subsystem 性能から定義され，Main Memory 容量と Memory bandwidth，Memory latency および Cache size（L2 Cache）が支配する．ここで，動作アプリケーションまたはデータを充足する Main Memory 容量が与えられ，かつ高帯域の Memory bandwidth を持つ場合，Main Memory から L2 Cache に何回メモリデータ供給が起こるかという命題に置き換えられる．しかるに，その回数分の Memory latency が空間計算量（space complexity）の時間コストとなる．次に時間計算量（time complexity） については，アプリケーションの演算 CPU クロック数と演算器（Arithmetic Logic Unit）へデータが到着するまでのトラベルタイム（L2 Cache latency）で定義される．ここで演算 CPU クロック数は，高度な Micro Architecture が備わった CPU（RISC プロセッサ）の場合，先行制御とパイプライン処理や Multi Core・Multi Thread の並列実行により，見かけ上の演算 CPU クロック数は縮退する．よって，オリジナルの計算量 *f(n)=Onlog(n)*との積が時間コストになる．尚，Database の総項目数(Elements)は *n* である．

3.4 ハードウェア・プラットフォームの選択

IEDB System 構築に向けて，その性能を支えるコンピューティングシステムを選択することは重要である．しかし，既往の Database 関連論文では，この部分を多角的・定量的に分析した論文は少ない．IEDB System のハードウェア・プラットフォーム選択における要件は，3 章 3.2 節で示した①～④の要素に加えて，ランニングコスト低減の観点から⑤低い消費電力も重要な要素である．文献[5]で Hubner らは Apple Silicon SoC のユニファイドメモリアーキテクチャ（Unified Memory Architecture）とその可能性に関して，Memory Subsystem と演算処理機能を中心に包括的な評価データを提供している．特に注目するポイントは，Memory bandwidth と Cache size，データレイテンシー（Memory latency と L2 Cache latency）の仕様である．さらに，IEDB System のランニングコストに影響する実運用時の消費電力評価では，同一世代の Intel プロセッサと比較し，1/3 以下の消費電力で同等以上の CPU 処理性能を提供している点である．

また，文献[6]において，Zhang らは Apple Silicon SoC の CPU（RISC）の Micro Architecture は，非常に優れたキャッシュ構成と大きな容量を持ち，効率的な RISC 命令パイプラインを備えていると結論付けている．

以上のことより，MongoDB DBMS で構成した IEDB System は，Apple Silicon M2 Pro SoC をエンジン（Engine）として採用することが最適であると判断した．Table 1 にこの SoC の諸元[7]を示す．

Table 1. Specification of Apple Silicon M2 Pro SoC [7].

Apple M2 Pro SoC	Value
Frequency	3.5GHz
Total Cores	12
Total Threads	12
L2 Cache Size	32MB (shared)
L2 Cache Latency*	≈10 ns
Main Memory Size	32 GB
Main Memory Latency*	≈100 ns
Main Memory Bandwidth	200 GB/s (LPDDR5-6400)
Thermal Design Power (TDP)	30 W

*Estimated from benchmarks and modern processor design specifications. (e.g., AMD Ryzen)

4. 提案方式と実装

4.1 実験プロセスにおけるデータ収集と統合データベースの連携関係

実験データを IEDB System の MongoDB へデータ収集する方法においては，実験プロセスのフロー図を定義し，各工程単位でどのようなデータがどのような構成の集合として記録されるのか認識することが重要である．これは，実験条件や測定・解析データの依存関係を把握し，どのような多次元データ連携になっているのか明確化することでもある．このことで，共通 API 作成の効率も上がる．本論文の提案では，フッ化物のコンビナトリアル(Fluoride Combinatorial)系における実験プロセス・フローを代表例として概説する (Fig.4)．その他のグループの実験に関しては，必要に応じて Fig.2 と対比しながら説明する．

Fig.4 左側レシピ（Recipe）は，電極や電解質の薄膜合成に用いるウエハ（Wafer）の基板材料と合成に用いるターゲットの種類とスパッタリング（Sputtering）による成膜条件などが Recipe シート（Excel）に記録され，その条件に従い第一段階の Sputtering 装置により Wafer 上に成膜される．この時，電極や電解質は，別 Wafer として作製される．次の実験ステップでは，正極・負極の活物質（Active materials）の条件を Wafer 上のセル（Cell）毎に振りながら Sputtering し，元素を積層（Layer stack & measurement）する．この実験における条件と実験装置による測定結果が Recipe とは別シートに記録される．ここで，実験装置は IoT による自動計測となっており実験装置付属の PC からローカルデータ収集 NAS（Network Attached Server）へ生データ（Raw Data）が集められる．さらに，成膜処理された Wafer 上の元素分布とその同定のため，蛍光 X 線分析装置（X-ray Fluorescence：XRF）を用いた計測と X 線回折装置（X-ray Diffraction：XRD）または SPring-8（Synchrotron XRD）により結晶構造の同定をおこなう．これらの計測データは，Wafer あたり 181 個の Cell 毎の座標データに対する測定値となっており，最初の実験サンプルの合成条件を記録した Recipe から Cell へデータの 3 次元拡張が生じる．また，この実験段階では一部のサンプルに対して Cell 毎の活性化エネルギーと導電率を測定し記録する．最終段階の実験では，Battery 性能測定（Battery perf. measurement）をおこなう．電極・電解質と区分して加工した Cell に対して，複数条件の組み合わせをおこない電気化学測定（Electrochemical）を実施する．この測定では基本的電池性能評価のために EIS（Electrochemical Impedance Spectroscopy）や CV（Cyclic voltammetry），CD（Charge discharge）などの項目を測定し，電気伝導率では，Arrhenius 解析から活性化エネルギーを算出する．前述の実験条件シート，測定結果データ，および解析データは，IoT や実験グループ内の Network を通じてローカルデータ収集 NAS へ一旦集約する．最後の Battery perf. measurement において，Cell の切り出しと，固体電解質と電極の異なるサンプル Wafer 同士の組み合わせが生じることから，データ連携構造は，第 2 段階の Layer stack & measurement の 3 次元拡張からより複雑な次元拡張が発生する構図になっている．MongoDB DBMS のデータ管理構造は，Database → Collection → Documents という階層関係による管理の枠組みを提供している．Fluoride Combinatorial 系の Database では，Collection にデータの次元が拡張される区切りで当てはめる．Fig.4 を例にすると Recipe，Layer stack & measurement，Battery perf. measurement のシート単位に Collection に当てはめることで，シートに記録されている Wafer サンプルと連携した Cell サンプルが Documents として収集される．さらに，Documents に収集されたそれぞれ

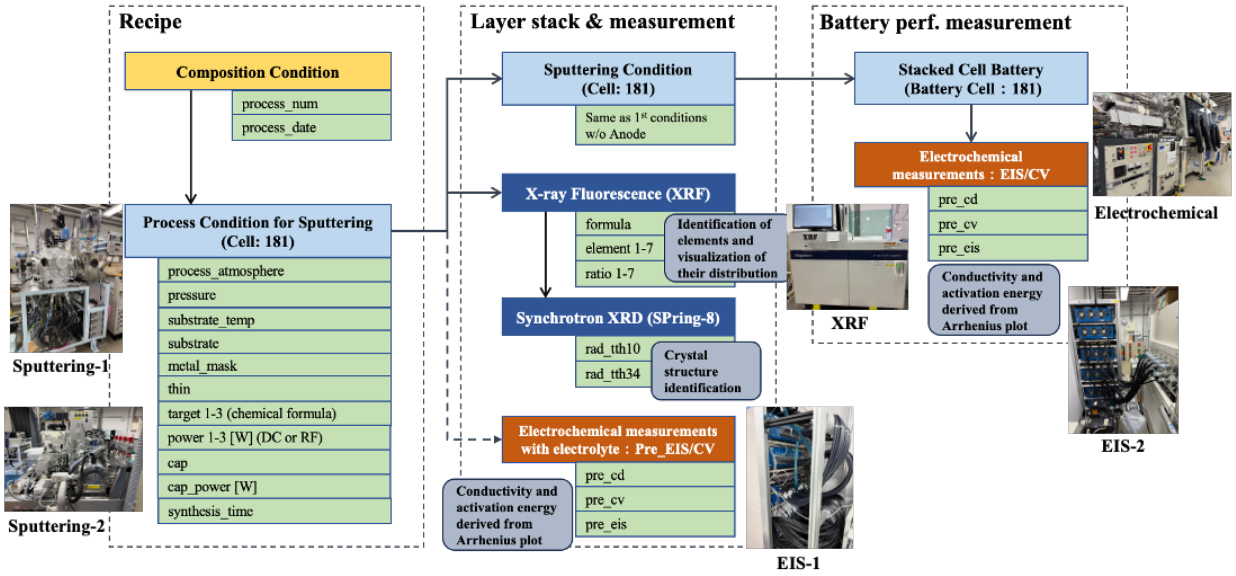


Fig. 4. Example of experimental and analytical process flow on fluoride combinatorial.

の Wafer，Cell，および Battery が条件・測定・解析の項目別データの要素（Elements）を持つことになる．また，Wafer サンプルから Cell サンプル, Battery とのデータ相関を明確化するためにそれぞれのサンプルは認識番号（ID）を自動的に付与する．このことで，共通 API からのサンプル間の相関関係の認識が容易くなる．Fig.2 に図示した Sulfides，Oxides，Fluorides に関しても同様に，Materials の元素種，秤量，粉砕・混合（Milling），焼成（Sintering），ペレット形成（Pelletizing）とその実施環境の合成条件を記録した Recipe の集合データと，XRD の計測・解析結果，および EIS・Battery の測定・解析をそれぞれ Mongo DB Collection としてグループ個別の Database に収集した．

4.2 統合実験 Database System のアーキテクチャと機能

Fig.5 は，各実験グループの Database を MongoDB DBMS に収納し，IEDB System を構成した Block Diagram である．前節で述べたデータ収集方式に基づいて，それぞれの Database は図で示すように Collection とその Documents を階層管理する方式としており，新規の Database や Collection，Documents の追加にも柔軟な対応が可能で，スケーラビリティが高い．さらに，この MongoDB DBMS は，共通 API によりデータの読み書き（Read/Write），検索（Search）などの Transaction が可能である．また，当該 API は，グループ・ユニークなデータシートのフォーマット差異を吸収することが可能である．次に，利用者が IEDB System へ容易にアクセスする Dashboard 機能として Web System をフロントに構築し，Network を介して検索と集計などの操作が実行できる GUI 画面を提供している．MI との連携機能に関しては，Data Frame と呼ぶ機能を提供しており，Pandas や SQLite 等へ Dataset を取り出せることから，ML や Computational Simulation との連携実行が可能である．IEDB System 運用の観点から，データのアップデートなどの Data Management に関しても，オープンソフトウェアの MongoDB Compass を採用することで容易に当該 DBMS の維持管理作業が行える．IEDB System にデータ集積した結果を Table 2 に示す．トータルのサンプル数は, 45,031 件であり, その項目数（Item Counts）は 30,998,114 個，総ファイル容量は，412.9MB であった．尚，画像ファイルなどはポインターとして別フォルダーに格納しているため，このファイル容量に含まれていない．

Table 2. Scale of IEDB System.

Database Name	Sample Quantity	Item Counts	File Size [MB]
Sulfide	4,567	20,181,580	228.4
Oxide	3,581	4,721,756	55.6
Fluoride	1,881	5,148,100	54.6
Combinatorial Fluoride	35,002	946,678	74.3
Sum	45,031	30,998,114	412.9

4.3 計算量理論による Database System の性能推定

IEDB System の性能ポテンシャルを推定する．目標値は，全ての Database に記録した全項目を検索する性能を指標とした．指標値は，IEDB System 利用ユーザが検索・閲覧に際してストレスを感じさせない待ち時間を担保するため，検索条件項目が 10 個の時に全要素（Elements）の検索に要する時間を約 1 秒以下とした．

推定値を導出する計算根拠の論理的な詳細フローを Fig.6 に示す．当該計算では，3 章 3.3 節で発想した検索計算量を決定するアルゴリズム Eq.(4)を用い, RISC プロセッサのパイプライン処理による L2 Cache Memory・Memory Subsystem のオーバーヘッド相殺と動作コア数（Table 1，Fig.3）の並列実行効果を考慮することで推定値を導出した．IEDB System に記録した約

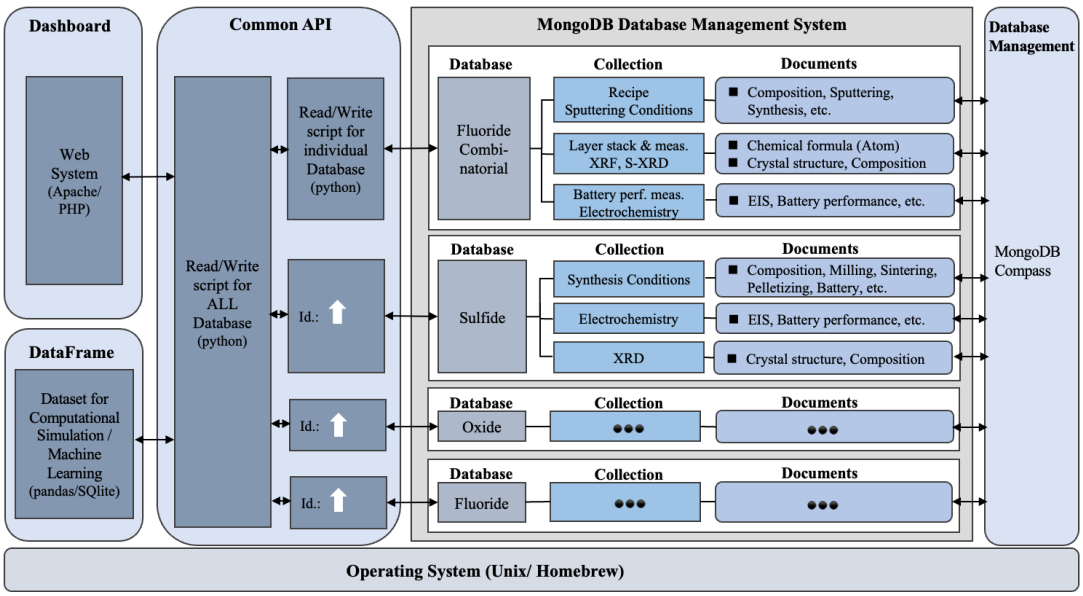


Fig. 5. Block Diagram of IEDB System.

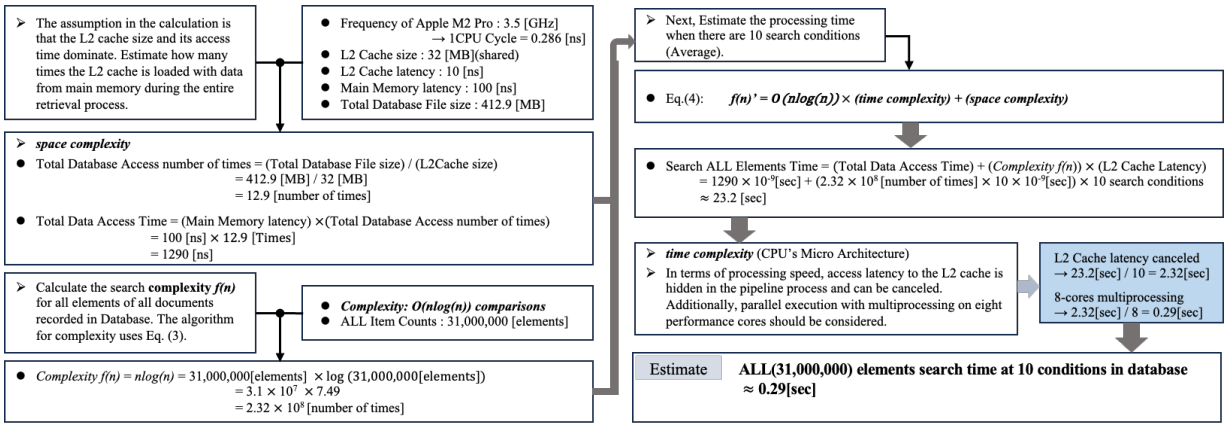


Fig. 6. Estimating search performance of IEDB System using Complexity Order Growth Theory.

3,100 万個の要素に対し，10 個の検索条件による全検索推定値は，0.29 [sec]となり，目標値をクリアできるという見積もりを導出した．

4.4 Low-Cost なハードウェア・プラットフォーム

産業応用への適用では，開発・実装コストと運用コストを如何に低減させるかということが非常に重要視される．IEDB System 実現に際しても当該課題を最重要として位置付けた．IEDB System を実現させた低消費電力サーバは，低価格で高性能な Apple Silicon M2 Pro SoC 搭載の Apple Mac Mini とした．当該 Mac Mini の諸元は，筐体サイズ（H : 3.58cm，W : 19.7cm，D : 19.7cm），重量 : 1.28Kg，消費電力 : 185W であり，非常にコンパクトで設置スペースの自由度が高い．

5. 有効性検証

本論文の有効性検証においては，提案方式の基礎とした DBMS 性能推定値が実測値を満たしているかという観点と，データ駆動型材料探索（MI）に活用したケースにおいて共通 API と連携した Data Frame 実装が有効に機能したかという観点の両面から検証をおこなう．

5.1 評価方法

先ず，IEDB System に対して，性能検証を実施する．性能検証では，4 章 Fig.5 で示した Sulfides, Oxides, Fluorides, Fluoride Combinatorial，各々の Database と全ての Database への検索実行時間を MongoDB に用意された Pymongo ライブラリの find_one 検索関数を使用し，search 開始から search 終了までの時間を測定するということで実行させた．この時，検索条件は 10 個とし，必ず全ての項目を検索するように存在しない条件を与えた．

Table 3. Actual IEDB System search processing time.

Database Name	Sample Quantity	Item Counts	Execution Time [sec]
Sulfide	4,567	20,181,580	0.058
Oxide	3,581	4,721,756	0.043
Fluoride	1,881	5,148,100	0.036
Combinatorial Fluoride	35,002	946,678	0.108
All Database	45,031	30,998,114	0.223 (estimate:0.29)

このことで検索動作は，最初の Element から最後の Element までをアクセスする．検証結果を Table 3 に示す．実際の全体 Database 検索実行時間は，0.22 [sec]であった．これは推定値の 0.29 [sec]に近似した値であり，1 秒以下という目標値も満たし，非常に高速な Transaction を可能とする．

5.2 計算・機械学習との連携結果

次に，IEDB System の MI 連携機能に関する有効性を検証する．当該検証では，Sulfides と Oxides の Database を用いて，その活用による研究成果例を挙げることで代用する．

Fig.7(a)では，高伝導率を示す硫化物系固体電解質の合成条件を見出すことを目的に，Database 中の電解質サンプルについて組成中の Li（リチウム）および S（硫黄）重量比と伝導率をマッピングしている[8]．さらに Fig.7(b)は，結晶構造情報を含む XRD データを次元削減してできるクラスタについて，その構造と伝導率を共に示すことで，XRD データに含まれる特徴量活用の有効性が確認された[8]．次に Fig.8(a)，(b)は，組成と伝導率

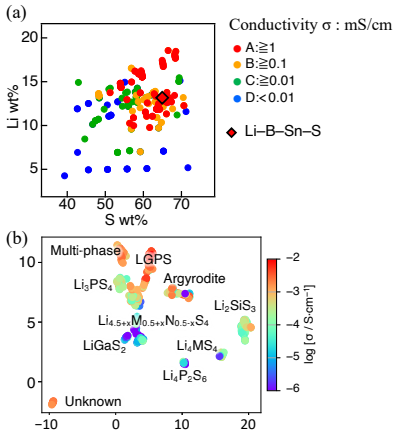


Fig. 7. Database mapping using feature value [8].
(a) Relationship between compositional feature value and conductivity. (b) Relationship between XRD and conductivity. Reproduced with permission of The Electrochemical Society of Japan.

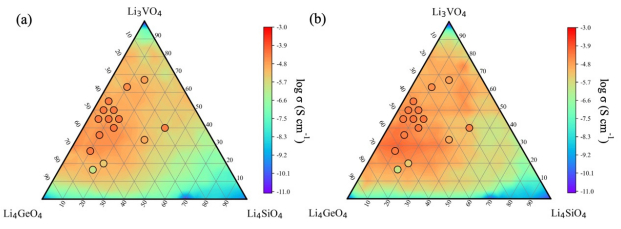


Fig. 8. Ionic-conductivity predications by the (a) NGBoost and (b) LightGBM models and experimental results for quasi-ternary Li_3VO_4-Li_4GeO_4-Li_4SiO_4 [9]. Fig. 8, by The Electrochemical Society of Japan, Licensed under CC BY 4.0.

のデータから複数の ML アルゴリズム（Random Forest, NGBoost, LightGBM など）を用い，それらアルゴリズムによる計算で求めた予測値と実験値との相関比較から，伝導率予測につなげる研究事例である[9]．このように， どのような特徴量がイオン伝導率と相関するファクターなのか，どのような ML アルゴリズムを用いて算出した結果がより現実に則したものなのかどうか，実験データとの比較において初めて MI の有効性も確認できることになる．

以上，複数の研究事例[8，9]の成果により，IEDB System はデータ駆動型材料探索（MI）との連携においても有効に機能することを確認した．

6. 考察

6.1 Database 性能

計算量の理論から導出した検索に要する時間の推定値 0.29[sec]と実測値 0.22[sec]の乖離に関して Database へのアクセスコスト概念と CPU（RISC）の Micro Architecture および Memory Subsystem のアーキテクチャから考察を与える．要因分析：計算量オーダー(order of growth)という概念のアルゴリズム Eq.(4)では，総データ量(Elements）が支配的である．しかし，この Transaction を逐次的に実行する場合，Fig.6 の計算フロー途中結果「Search ALL Elements Time ≈ 23.2 [sec] 」となる．他方，RISC プロセッサのパイプライン処理（Fig.9）とデータプリフェッチにより L2 Cache のアクセス時間は隠蔽することが可能で計算上は無視できる．文献[10]では，RISC プロセッサの性能指標を評価対象となるプログラムの実行時間 T は，PC（実行命令数），CPI（平均命令実行サイクル数），MC（CPU Cycle）の積で表現

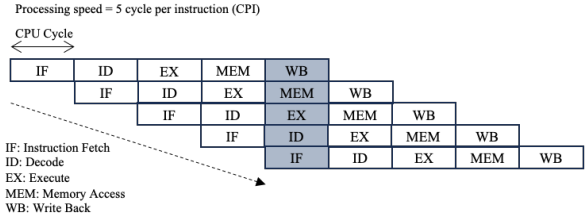


Fig. 9. Overview of basic RISC processor pipeline processing.

され，T が小さいほど高性能であるとしている（Eq.(5)）．

Fig.9 より逐次処理のケースでは CPI=5 であるが，パイプライン処理によりキャッシュミスのない理想的な処理では限りなく CPI=1 に近づく．さらに，複数 Core による並列実行をおこなうことによりさらに見かけ上の CPI は下がる．

$$T = PC \times CPI \times MC \quad \cdots\cdots(5)$$

以上のことから，推定値導出では，L2 Cache latency=10[ns]が隠蔽されることを考慮し， Search ALL Elements Time ≈ 23.2 [sec] を 1/10 の ≈ 2.32 [sec] とし，さらに 8 Core の効果を加味することで，1/8 の≈ 0.29 [sec] とした．ここで，プログラムの並列実行を 8 Core 動作と仮定しているが，これは Apple Silicon M2 Pro プロセッサの Performance Core 数が 8 個であることによる．もし，Efficient Core 4 個の動作も加えるなら，さらに短縮できる．また，当該概算では 12 個の Thread 動作効果を考慮していないが，これについても考慮することで低減が可能であり実測値 0.22[sec] に近似する．しかし，現実の動作では，オペレーティングシステム（OS）などが管理する複数プロセスがバックグラウンド動作することから Performance Core 動作に限定した処理とすることが妥当である．以上のことから，前述の乖離に関しては，オーダーの議論としてバランスしていると結論付けできる．

以上のことより，MongoDB DBMS を採用した IEDB System においては，Engine として採用する RISC プロセッサ（CPU）は，3 章 3.3 節で示した①～④の要素に加え，Micro Architecture による高度なタスク管理も検索 Transaction の実行時間を支配する要因であると考える．

6.2 Database の規模と機能進化

前述の通り，IEDB System に記録・保存（Archive）した実験データ量は，実験サンプル数が約 45,000 件であり，これら実験サンプルデータに含まれる項目数は，約 3,100 万個に上る．当該規模は無機材料 Database の中でも大きなサイズであり，全固体電池分野という特定分野では最上級の規模である．他の材料系 Database と規模とその内容を比較すると，結晶構造のデータを Archive する ICSD（Inorganic Crystal Structure Database）が約 30 万件，主に電子状態計算 DFT（Density Functional Theory）のデータを Archive する MP（Materials Project）[11]が 17 万件，材料の組成・構造・物性・電子構造の計算結果を Archive する NIMS（国立研究開発法人 物質・材料研究機構）の RDE[12]が 85 万 6 千件の規模である．それぞれの Database は専門分野の正確なデータを提供しており，材料研究開発の分野では，非常に重要な役割を担っている．ところで，本論文で方式と実装提案をおこなった IEDB System に Archive したデータは，ML や DFT 計算から算出された仮想の材料データではなく，失敗も含めた試行錯誤の実験結果から生み出された記録であり，新しい発見につなげる洞察を可能にする鳥瞰的な視点を提供する．さらに，DFT や ML を駆使して算出した仮想データが正しいかどうかの

検証をするための最終フェーズゲートでもある．このことにおいても極めて高い価値があると考えている．

6.3 産業応用分野への展開

IEDB System は，デジタル（Database）技術と全固体電池実験技術を融合した領域の産物であり，単一の技術領域だけの視点からシステムとして構成させることは極めて困難である．特に重要な要素は，実験手法と計測データの意味を咀嚼した上で実験プロセス・フローを策定し，そのプロセスの区切りとデータの相関を Database 構造へ反映させていくことにある．これは，製造工程などの産業 DX に必要な Database 開発においても同様であると類推できる．このように複数領域の技術開発要素を融合させていくことは，生産工程の自動化（Automation）や自律化（Autonomous）の実現を加速し，労働力不足を補う有用な手段となる．このように，産業 DX における Database 活用は，今後も大きく進展することが期待される．

7. 結論

全固体電池の研究開発に実装する「統合実験データベース基盤（Integrated Experimental Database System：IEDB System）」のデータ収集・管理アーキテクチャの提案をおこない，そのシステム開発の成果から，以下の 4 点の効果を確認した．

① 各実験グループ・ユニークな実験条件と測定・解析結果の高次元・非構造化データを IEDB System へ収集する方式として，グループ毎に実験プロセス・フロー図を定義し，それらの工程から出力されるデータを MongoDB DBMS 内の区分 Database とし，その区分した Database 内の Collection へストレートに持ち上がるデータコンバージョン方式が効率的であることを確認した．さらに，これらの区分 Database を共通 API で Wrapper することで，単一の仮想 Database を実現し，一元的な管理と Transaction（Web からの閲覧・検索，および Computational Simulation・Machine Learning 機能への Dataset 抽出）の実行が柔軟かつ効率的におこなえることを確認した．

② IEDB System の主要な Transaction 性能推定に際しては，計算量理論（Complexity Order Algorithm）を基礎とし，RISC プロセッサの Micro Architecture 性能と Memory Subsystem 性能から時間計算量（time complexity）と空間計算量（space complexity）を論理的に積み上げ融合させた新しいアルゴリズムを発想し適用した．IEDB System の Engine に採用した Apple Silicon M2 Pro SoC を用いた実証実験の結果，10 個の検索条件による全項目検索の Transaction 実行時間は，0.22[sec]であり，新規性能推定アルゴリズムが計算した数値 0.29[sec]とオーダー的にバランスした．これらにより新しい性能推定アルゴリズムの有用性を確認した．

③ IEDB System に収録した実験データ量は，サンプル数：45,031 件，項目数：30,998,114 個であり，全固体電池材料の実験 Database として最上級のデータ量を収納した．

④ データ駆動型材料探索（Materials Informatics：MI）との連携機能性に関しては，共通 API を通じた連携機能の有用性を確認した．

以上，本論文では，全固体電池研究向け Low-Cost 統合実験 Database を開発し，その応用社会実装の可能性を考察した．

尚，IEDB System は，2022 年後半より動向調査および方式検討に取り組み，2023 年後半にシステム開発をスタートさせ運用開始は 2024 年 4 月からであった．さらに，開発コストに関しては，一般的なシステムインテグレーション（SI）費用と比較し，1/20 で実現させた．

謝　辞

本開発の推進にあたり，多方面のサポートと機会を頂戴した東京科学大学 池松正樹特任教授に厚くお礼を申し上げます．

本開発の成果は，新エネルギー・産業技術総合開発機構（NEDO）2021～2025 年度「電気自動車用革新型電池開発 RISNG3」（JPNP21006）により得られたものです．また，著者の一人（S. H.）は JSPS 科研費 JP25H01962 および JP25K08787 の助成に感謝申し上げます．

参考文献

[1] K. Fujii: “Data Analysis Infrastructures in Plasma Experiments”, *Journal of Plasma and Fusion Research*, Vol.95, No.5, pp.206-207 (2019-5)

[2] R. Yamazaki, S. Sakata, and H. Urano: “Data Acquisition and Analysis Infrastructure in JT-60SA: Database Structure in JT-60SA”, *Journal of Plasma and Fusion Research*, Vol.95, No.5, pp.213-215 (2019-5)

[3] J. Cieslewicz, and K. A. Ross: “Database Optimization for Modern Hardware”, *Proceeding of the IEEE,* Vol.96, No.5, pp.863-878 (2008-5)

[4] 戸田誠之介：「日本数学会 1998 年度年会・企画特別講演：計算量理論における研究課題」, 1998 年 1998 巻 Spring-Meeting 号, pp.26-34 (2010-7: J-STAGE 公開日)

[5] P. Hubner, A. Hu, I. Peng, and S. Markidis: “Apple vs. Oranges: Evaluating the Apple Silicon M-Series SoCs for HPC Performance and Efficiency”, arXiv:2502.05317 (2025-2)

[6] Z. Zhang: “Analysis of the Advantages of the M1 CPU and Its Impact on the Future Development of Apple”, *20212nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE), 24-26 September 2021*, IEEE Xplore, (2022-2)

[7] Apple unveils M2 Pro and M2 Max: next-generation chips for next-level workflows, https://www.apple.com/newsroom/2023/01/apple-unveils-m2-pro-and-m2-max-next-generation-chips-for-next-level-workflows/

[8] S. Hori, and T. Nakamura: “A Database of Experiments for the Development of Sulfide-Type Solid Electrolytes”, *The Electrochemical Society of Japan*, Denki Kagaku Vol.92, No.4, pp.325-329 (2024-12)

[9] Y. Iwamizu, et al.: “Chemical Composition-Driven Machine Learning Models for Predicting Ionic Conductivity in Lithium-Containing Oxides”, *The Electrochemical Society of Japan*, Advanced online publication, No.2, pp.1-8 (2025-3)

[10] K. Matsubara: “Explanation of Cache Technologies for RISC Processor”, *Information Processing of Japan, IPSJ Magazine*, Vol.36, No.5, pp.454-458 (1995-5)

[11] M. K. Horton, S. Dwaraknath and K. A. Persson: “Promises and Perils of computational materials database”, *Nature Computational Science*, Vol.1, pp.3-5 (2021-1)

[12] National Institute for Materials Science NIMS NOW: “Materials DX: fundamentally transforming materials R&D”, Vol.19, No.5 (2021-12)

付　録 A（Appendix A）

A1. データモデル

A1.1 主要 Collection のスキーマ定義（JSON Schema）

固体電解質サンプルの合成条件および物性値を管理する Fluoride DB Sample Collection のスキーマ定義例（抜粋）を Fig. A.1 に示す．次に，X 線回折（XRD）測定データを管理するコレクションのスキーマ定義例を Fig. A.2 に示す．Histogram 配列は通常 2,000–5,000 点のデータポイントを含み，平均ドキュメントサイズは約 150 kB である．

```
    {
      "$jsonSchema": {
        "bsonType": "object",
        "title": "Sample Document Schema",
        "required": ["sample No", "chemical formula", "date"],
        "properties": {
          "_id": {
            "bsonType": "objectId",
            "description": "MongoDB自動生成ID"
          },
          "No": {
            "bsonType": "int",
            "minimum": 1,
            "description": "通し番号（整数、1以上）"
          },
          "date": {
            "bsonType": "int",
            "minimum": 20000101,
            "maximum": 29991231,
            "description": "実験日（YYYYMMDD形式、整数）"
          },
          "sample No": {
            "bsonType": "string",
            "pattern": "^[A-Z]{2}[0-9]{2,4}$",
            "description": "サンプル番号（例: TO05、FL123）"
          },
          "chemical formula": {
            "bsonType": "string",
            "minLength": 3,
            "description": "化学式（例: Li3YCl6）"
          },
          "mixing method": {
            "bsonType": "string",
            "enum": ["Planetary ball mill", "Vibration mill", "Manual mixing",
"Other"],
            "description": "混合方法"
          },
          "rotation speed [rpm]": {
            "bsonType": ["double", "null"],
            "minimum": 0,
            "maximum": 10000,
            "description": "回転速度（rpm、0–10000）"
          },
          "mixing time [h]": {
            "bsonType": ["double", "null"],
            "minimum": 0,
            "description": "混合時間（時間）"
          },
          "sample height [cm]": {
            "bsonType": ["double", "null"],
            "minimum": 0,
            "description": "サンプル高さ（cm）"
          },
          "total log": {
            "bsonType": ["double", "null"],
            "description": "対数導電率 log₁₀(σ / S cm⁻¹)"
          },
          "total log(A/S cm-1K)": {
            "bsonType": ["double", "null"],
            "description": "対数プレエクスポネンシャル因子"
```

Fig. A.1. Fluoride DB Sample Collection JSON Schema.

A1.2 代表的ドキュメント例

Fig. A.3 は Sulfide DB Electrochemical Collection のドキュメント例（抜粋）である．また，Fig. A.4 は Sulfide DB Synthesis Collection のドキュメント例（抜粋）である．

A1.3 スキーマバリデーションルール

MongoDB のスキーマバリデーション機能を用いてデータ整合性を確保する．バリデーションルール設定例として，Fig. A.5 に Fluoride DB Sample Collection の設定を示す．バリデーションレベルは，(i) moderate : 既存ドキュメントの更新時のみ検証，（ii）strict : すべての挿入・更新時に検証，の 2 段階である．バリデーションアクションは，（i）warn : 警告ログを出力し挿入を許可，（ii）error : バリデーション失敗時に操作を拒否，とした．参照整合性については，MongoDB が外部キー制約を提供しないため，アプリケーション層で管理する．

```
    {
      "$jsonSchema": {
        "bsonType": "object",
        "title": "XRD Document Schema",
        "required": ["SampleName", "histgram"],
        "properties": {
          "_id": {
            "bsonType": "objectId"
          },
          "SampleName": {
            "bsonType": "string",
            "description": "サンプル名（Sample.sample Noへの外部キー）"
          },
          "log": {
            "bsonType": ["string", "null"],
            "description": "測定ログ（装置情報、測定条件等）"
          },
          "Start": {
            "bsonType": "double",
            "minimum": 0,
            "maximum": 180
            "description": "開始角度（2θ、度）"
          },
          "Stop": {
            "bsonType": "double",
            "minimum": 0,
            "maximum": 180,
            "description": "終了角度（2θ、度）"
          },
          "Step": {
            "bsonType": "double",
            "minimum": 0.001,
            "maximum": 1.0,
            "description": "ステップ幅（度）"
          },
          "histgram": {
            "bsonType": "array",
            "minItems": 100,
            "items": {
              "bsonType": "array",
              "minItems": 2,
              "maxItems": 2,
              "items": {
                "bsonType": "double"
              }
            },
            "description": "XRDパターン [[2θ, Intensity], ...]"
          }
        }
      }
    }
```

Fig. A.2. Fluoride DB XRD Collection JSON Schema.

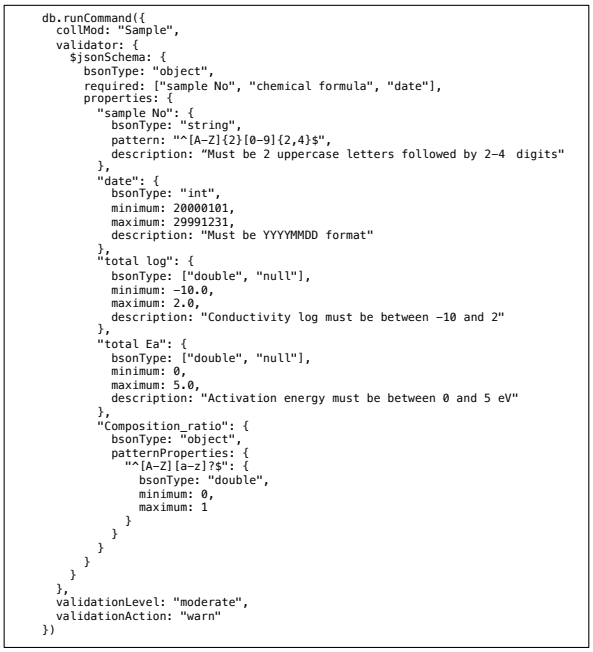
```
{

  "_id": ObjectId("507f1f77bcf86cd799439014"),
  "RegistrationDate": "2023-03-15",
  "MeasurementDate": "2023-03-14",
  "MeasurementSampleName": "S_Li3PS4_001",
  "MeasurementFileName": "S_Li3PS4_001_RT.z",
  "PressureReceivingArea": 0.785,
  "PelletThickness": 1.2,
  "AppliedPressure": 360.0
  "TotalConductivity": 1.6e-4,
  "ActivationEnergy": 0.33,
  "FittingData": {
    "R_grain": 125.3,
    "R_grain_boundary": 387.2,
    "CPE_n": 0.82,
    "Chi_squared": 1.2e-4
  },
  "ImpedanceData": [
    [1000000.0, 512.5, -12.3],
    [794328.0, 510.8, -15.7],
    ...
    [0.1, 8523.4, -4521.8]
  ],
  "Note": "Room temperature (25°C) measurement"
}
```

Fig. A.3. Sulfide DB Electrochemistry Collections Document.

```
{
      "_id": ObjectId("507f1f77bcf86cd799439013"),
      "RegistrationDate": "2023-03-10",
      "RegistrationTime": "14:35:22",
      "FormatVersion": "3.0",
      "SheetFileName": "synthesis_log_2023Q1.xlsx",
      "SampleName": "S_Li3PS4_001",
      "Prefix": "S",
      "Suffix": "001",
      "Formula": "Li3PS4",
      "Composition": {
        "Li": 3.0,
        "P": 1.0,
        "S": 4.0
      },
      "Composition_ratio": {
        "Li": 0.375,
        "P": 0.125,
        "S": 0.5
      },
      "MixingMethod": "Planetary ball mill",
      "MixingDevice": "Fritsch Pulverisette 7",
      "MillingTime": 20.0,
      "MillingSpeed": 370.0,
      "FiringMethod": "Tube furnace",
      "FiringTemperature": 550.0,
      "FiringTime": 6.0,
      "FiringAtmosphere": "H2S",
      "Quenching": "Yes",
      "QuenchingMethod": "Liquid nitrogen",
      "TotalMass": 2.5,
      "Yield": 92.3,
      "ColorBefore": "White",
      "ColorAfter": "Pale yellow",
      "Density": 1.86,
      "Note": "β-Li3PS4 phase confirmed by XRD"
    }
```

Fig. A.4. Sulfide DB Synthesis Conditions Document.



```
db.runCommand({
  collMod: "Sample",
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["sample No", "chemical formula", "date"],
      properties: {
        "sample No": {
          bsonType: "string",
          pattern: "^[A-Z]{2}[0-9]{2,4}$",
          description: "Must be 2 uppercase letters followed by 2-4  digits"
        },
        "date": {
          bsonType: "int",
          minimum: 20000101,
          maximum: 29991231,
          description: "Must be YYYYMMDD format"
        },
        "total log": {
          bsonType: ["double", "null"],
          minimum: -10.0,
          maximum: 2.0,
          description: "Conductivity log must be between -10 and 2"
        },
        "total Ea": {
          bsonType: ["double", "null"],
          minimum: 0,
          maximum: 5.0,
          description: "Activation energy must be between 0 and 5 eV"
        },
        "Composition_ratio": {
          bsonType: "object",
          patternProperties: {
            "^[A-Z][a-z]?$": {
              bsonType: "double",
              minimum: 0,
              maximum: 1
            }
          }
        }
      }
    }
  },
  validationLevel: "moderate",
  validationAction: "warn"
})
```

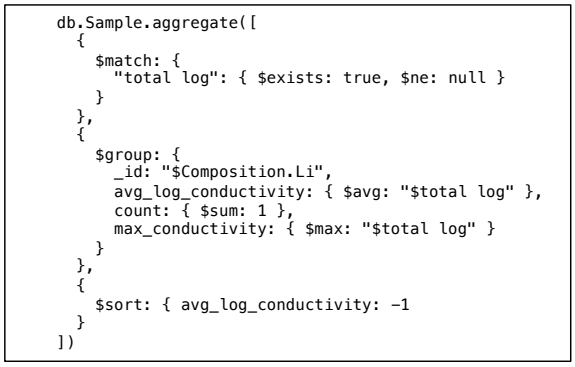
Fig. A.5. Validation Rule Settings for Fluoride DB Sample Collection.

A1.4 インデックス設計

本データベースシステムでは，(i) ユニークインデックス（主キー相当フィールド），(ii) 単一フィールドインデックス（頻繁に検索されるフィールド），(iii) 複合インデックス（複数フィールドによる範囲検索），(iv) 部分インデックス（条件付きインデックス）を採用する．これらの設計戦略を含む例として Fig. A.6 に Sulfide DB Synthesis のインデックス設計を示す．また，複雑な集計クエリの例として，「組成別平均導電率」を算出するクエリ例を Fig. A.7 に示す．

```
// 1. サンプル名ユニークインデックス
db.Syn.createIndex(
  { "SampleName": 1 },
  { unique: true, name: "idx_sample_name_unique" }
)
// 2. 化学式インデックス
db.Syn.createIndex(
  { "Formula": 1 },
  { name: "idx_formula" }
)
// 3. 焼成温度範囲検索用
db.Syn.createIndex(
  { "FiringTemperature": 1 },
  { name: "idx_firing_temp" }
)
// 4. 複合インデックス（焼成条件）
db.Syn.createIndex(
  { "FiringTemperature": 1, "FiringTime": 1, "FiringAtmosphere": 1 },
  { name: "idx_firing_conditions" }
)
// 5. 混合方法インデックス
db.Syn.createIndex(
  { "MixingMethod": 1 },
  { name: "idx_mixing_method" }
)
// 6. 日付インデックス（新しい順）
db.Syn.createIndex(
  { "RegistrationDate": -1 },
  { name: "idx_registration_date_desc" }
)
```

Fig. A.6. Sulfide DB Synthesis Index Design.



```
db.Sample.aggregate([
  {
    $match: {
      "total log": { $exists: true, $ne: null }
    }
  },
  {
    $group: {
      _id: "$Composition.Li",
      avg_log_conductivity: { $avg: "$total log" },
      count: { $sum: 1 },
      max_conductivity: { $max: "$total log" }
    }
  },
  {
    $sort: { avg_log_conductivity: -1
  }
])
```

Fig. A.7. Example of a complex aggregation query: Average conductivity by composition (Aggregation pipeline).

A2. 運用設計とセキュリティ

IEDB System の運用（Operation）設計を Table A.1 に示す．また，セキュリティ（Security）施策を Table A.2 に示す．さらに，データ・ガバナンス（Data Governance）方針を Table A.3 に示す．

Table A.1. Operation.

#	項目	内容
1	バックアップ/リストア	● 外付けSSDへ，毎日深夜にフルバックアップを取得する（mongodump） ● 併せて，macOSのTime Machineによりシステム全体の差分バックアップ（スナップショット）を取得する．mongodumpと実行時間が重複しないようにスケジュールし，稼働中DBのディレクトリはバックアップ対象外とする
2	レプリカセット	● Secondaryノードへの複製同期を行う．Primary障害時の復旧はバックアップからのリストアを基本とし，フェイルオーバーは構成しない
3	障害シナリオとRTO/RPO	● Primary障害時の復旧時間目標（RTO）は最大4時間とする．データ復旧点目標（RPO）は最新バックアップ時点とし，障害部位に応じて最適な復旧手順（交換/復旧）を適用する
4	監視メトリクス	● 性能劣化，ディスク使用率等の稼働状況を常時監視し，異常を自動通知する ● アクセスログ（不正アクセス兆候，データダウンロード量等）を記録し，必要に応じてメールで通知する
5	スキーマ進化とマイグレーション	● DBごとにドキュメント内でスキーマ世代管理を行う．更新時はバリデーションプログラムによりデータ整合性を検証する

Table A.2. Security.

#	項目	内容
1	認証/認可	● ネットワーク層ではファイアウォールを配置し，VPN経由のSSH接続のみに制限する ● デバイスIDによる接続制限を設け，認証局が発行したクライアント証明書を導入した端末のみ接続を許可する ● MongoDB（アプリケーション層）ではユーザID/パスワード認証を用い，アクセス制御および監査ログ記録を行う
2	監査ログ	● ログインユーザ，アクセス対象，データダウンロード量等を監査ログとして記録し，異常検知時は自動通知する
3	データ秘匿化/匿名化	● データの完全性確保としてSHA-256によるハッシュおよびECDSA署名を用いた改ざんん検知を行う ● 匿名化は実施しない

Table A.3. Data Governance.

#	項目	内容
1	メタデータ	● 実験装置付属のIoT PCからNASへ、実験単位でデータを収集・保存する．併せて，実験者による実験ノート記録を併用する
2	プロベナンス	● 原材料→合成実験→サンプルデータ→派生データ（追試等）のデータ系譜を記録・保存する
3	バージョニング	● ドキュメントレベルのバージョン管理を行い，変更差分を随時記録して全バージョンを保持する
4	アクセス申請フロー	● DBアクセス権限は責任教員がシステム管理者へ申請し，管理者が(i)クライアント証明書(デバイスID証明書）の配布，(ii) DBログインID/パスワードの通知を行う

A3. API 概略仕様

Table A.4 は，IEDB System の API 仕様の代表的なエンド・ポイント（API End Point）をまとめたものである．代表的な API End Point は，大きく 6 種類あり「(a) サンプルの CRUD 操作」，「(b) XRD 操作」，「(c) インピーダンス・導電率操作」，「(d) 統計・分析」，「(e) バッチ処理」と計算・機械学習による材料探索：Materials Informatics（MI）を効率的に行うための「(f) MCP Tools の API」に分けられる．これらの API を複合し，研究者がインタラクティブに材料探索を進めるダッシュボードサンプル例を付録 C（Appendix C）に示す．

Table A.4. API design overview.

(a) Sample CRUD

#	Method	API End Point	Description
1	GET	/api/v1/samples/{sample_no}	サンプル取得
2	GET	/api/v1/samples	サンプル検索・一覧
3	POST	/api/v1/samples	新規サンプル作成
4	PUT	/api/v1/samples/{sample_no}	サンプル更新
5	DELETE	/api/v1/samples/{sample_no}	サンプル削除

(b) XRD操作

#	Method	API End Point	Description
1	GET	/api/v1/xrd/{sample_no}	XRDデータ取得
2	POST	/api/v1/xrd	XRDデータ登録
3	GET	/api/v1/xrd/{sample_no}/peaks	ピーク情報取得
4	POST	/api/v1/xrd/{sample_no}/analyze	XRD解析実行

(c) インピーダンス・導電率操作

#	Method	API End Point	Description
1	GET	/api/v1/impedance/{sample_no}	インピーダンスデータ取得
2	POST	/api/v1/impedance	インピーダンスデータ登録
3	GET	/api/v1/impedance/{sample_no}/arrhenius	Arrhenius解析
4	GET	/api/v1/conductivity/{sample_no}	導電率データ取得

(d) 統計・分析

#	Method	API End Point	Description
1	GET	/api/v1/statistics/conductivity	導電率統計
2	GET	/api/v1/statistics/composition	組成統計
3	GET	/api/v1/correlations/xrd-conductivity	XRD-導電率相関
4	POST	/api/v1/recommendations	サンプル推奨

(e) バッチ処理

#	Method	API End Point	Description
1	POST	/api/v1/batch/samples	複数サンプル一括作成
2	PUT	/api/v1/batch/samples	複数サンプル一括更新
4	POST	/api/v1/batch/export	データ一括エクスポート

(f). MCP Tools overview （MCP API Server）

#	Category	Number of tools	Main functions
1	XRD Analysis	3	Ternary diagram data, FWHM calculation, peak detection
2	Generic Database Operations	4	Database list, collection list, statistics, generic query
4	Thin-film Data	2	Thin-film sample search, recipe retrieval
5	Materials Project	2	Material search, material ID search
6	Impedance / Arrhenius	2	Impedance data retrieval and Arrhenius analysis

Notes: (i) The server provides 26 tools across 8 functional categories.
(ii) The Dashboard Launch category includes one recommended integrated UI and five deprecated individual UIs.
(iii) AI optimization tools use Bayesian optimization (PHYSBO) for material design.

付　録 B（Appendix B）

B1. IEDB System 性能評価結果と Computing System アーキテクチャの課題

以下に，IEDB System の Database 性能評価結果を示すとともに，同システムで使用したプロセッサ（Apple M2 Pro SoC）について，Database 性能評価パラメータのプロファイリング分析結果を報告する．

Table B.1–B.2 に，ベンチマーク環境および評価パラメータの概要を示す．続いて，代表的な操作別性能とスケーリング性能の比較結果を Fig. B.1–B.4 に示す．

次に，IPC/CPI の理論性能と各操作における実測値の比較を Table B.3–B.4 に示す．また，Table B.5–B.6 には，Database で一般的に想定されるキャッシュミス率と，当該 Database の SELECT 操作におけるキャッシュミス率の推定結果を示す．

スケーリング特性については，INSERT および SELECT 操作に対する同時接続数の影響を Table B.7–B.8 に示す．さらに，I/O 待ちによるオーバーヘッドについて，操作別の比較結果を Table B.9 に示す．

mongostat/mongotop によるベンチマーク結果は Table B.10–B.13 に整理し，それらを要約した結果と分析を Table B.14 に示す．

Table B.15 は，全固体電池研究用途の Database における操作別の利用頻度を評価した結果である．最後に，全体を通した性能評価結果およびプロファイリング分析結果を Table B.16–B.17 にまとめる．

Table B.1. Benchmark experimental environment.

Item	Specification
CPU	• Apple M2 Pro (12 cores: 8 performance + 4 efficiency, L2 Cache size: 32 MB (L2: 16 MB ＋SLC*: 16 MB)) , *System Level Cache
Main Memory	• 32 GB LPDDR5 (Unified Memory, Memory bandwidth: 200 GB/s)
Storage	• 8 TB SSD (NVMe)
OS	• macOS Sonoma 14.x
MongoDB	• 7.X (WiredTiger)
Python / pymongo	• 3.11.x / 4.x

Table B.2. Evaluation axes and experimental parameters.

Axis	Parameter	Value
CRUD operations	Operation type	• INSERT, SELECT (Simple/Range/Complex), UPDATE, DELETE
	Concurrency	• 1, 5, 10
	Dataset	• 1,000 operations / run
Indexing	Configuration	• none, basic (single-field）, full (compound)
	Target	• all CRUD operations
Parallelism	Concurrent connections	• 1, 5, 10
	Operation	• all operation types
Replica/Sharding	Configuration	• Single server (sharding not implemented)
Data volume	Size	• 1,000, 5,000, 10,000 documents

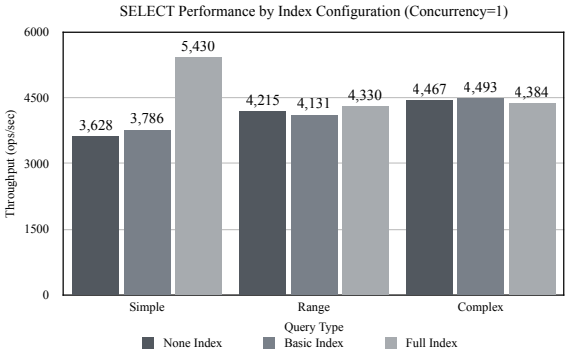


Fig. B.1. SELECT Performance.

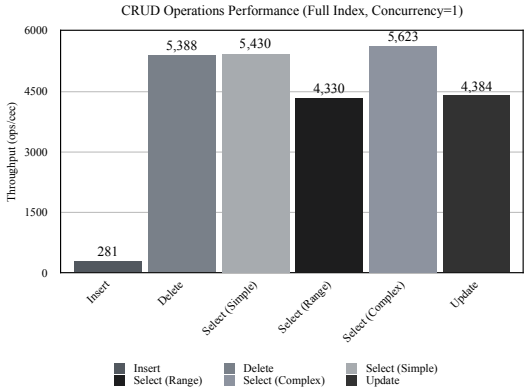


Fig. B.2. CRUD Operations Performance.

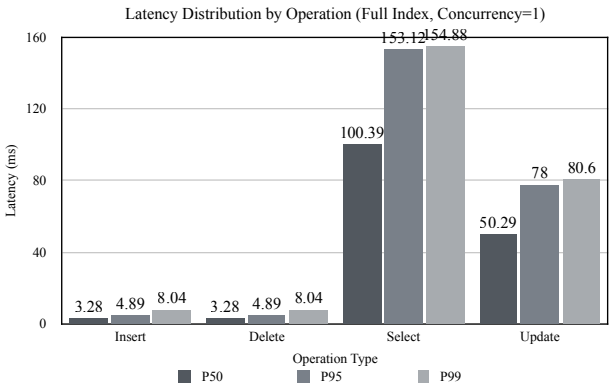


Fig. B.3. Latency Distribution by Operation.

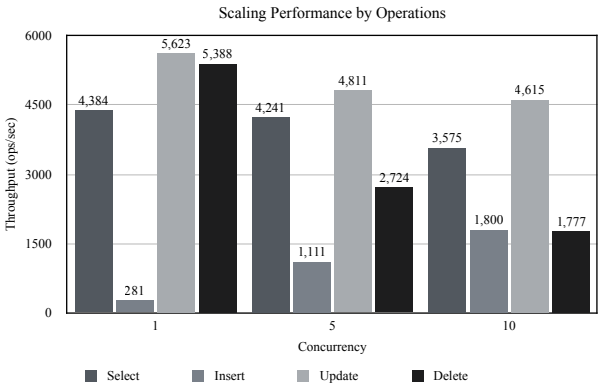


Fig. B.4. Scaling Performance by Operation.

Table B.3. Theoretical IPC/CPI of Apple M2 Pro cores.

Core type	IPC (Instructions/Cycle)	CPI (Cycles/Instruction)	Frequency
Performance core	6–8 IPC (peak)	0.125–0.167	3.5 GHz
Efficiency core	4–5 IPC	0.200–0.250	2.4 GHz

Table B.4. Estimated CPI by operation type (measured vs. theoretical).

Operation	Concurrency	Measured CPI	Theoretical CPI	Efficiency	Major bottleneck
INSERT (full index)	1	0.35	0.20	57%	Index update
INSERT (full index)	10	0.42	0.20	48%	Lock contention
SELECT (full index)	1	0.28	0.18	64%	B-tree traversal
SELECT (full index)	10	0.55	0.18	33%	Lock contention, cache misses
UPDATE (full index)	1	0.38	0.22	58%	Index update
UPDATE (full index)	10	0.48	0.22	46%	Lock contention

Table. B.5. Cache hierarchy and typical hit rates for database.

Cache level	Size	Hit rate	Miss penalty	Latency
L1 (data)	128 KB/core	60–70%	3–4 cycles	˜1 ns
L2 (cluster)	16 MB	85–90%	12–15 cycles	˜4 ns
SLC (system)	16 MB	95–98%	30–40 cycles	˜12 ns
Unified Memory	32 GB	100%	200–250 cycles	˜70 ns

Table B.6. Estimated cache miss rates by concurrency (SELECT).

Concurrency	L1 miss rate	L2 miss rate	SLC miss rate	Mean latency	Cache efficiency
1	35%	12%	3%	102.81 ms	High
5	42%	18%	5%	29.19 ms	Medium
10	48%	25%	8%	16.10 ms	Low

Table B.7. Measured parallel efficiency (SELECT, full index).

Concurrency	Measured throughput	Ideal throughput	Measured efficiency	Theoretical efficiency	CPU utilization
1	5430	5430	100.0%	100.0%	8.3%
5	4315	27151	15.9%	73.1%	41.7%
10	4577	54302	8.4%	50.9%	83.3%

Table B.8. Measured parallel efficiency (INSERT, full index).

Concurrency	Measured throughput	Ideal throughput	Measured efficiency	Speedup
1	281	281	100.0%	1.00×
5	1111	1406	79.0%	3.95×
10	1800	2812	64.0%	6.40×

Table B.9. Estimated I/O wait time by operation type.

Operation	Cache hit rate	Disk access frequency	I/O wait time	Impact on performance
INSERT	98%	Low (buffering)	< 1%	Negligible
SELECT	99%+	Very low	< 0.5%	Negligible
UPDATE	98%	Low	< 1%	Negligible
DELETE	99%	Very low	< 0.5%	Negligible

Table B.10. mongostat measurements (concurrency =1, SELECT).

Metric	Value	Description
query	∼5,400/sec	SELECT throughput
resident	1.2 GB	Resident memory size
mapped	0 MB	Memory-mapped files (not used by WiredTiger)
faults	0/sec	Page faults (none)
connections	10–20	Concurrent connections
network in	150 KB/s	Incoming traffic
network out	8 MB/s	Outgoing traffic

Table B.11. mongostat measurements (concurrency =10, SELECT).

Metric	Value	Description
query	∼4,600/sec	SELECT throughput (-15%)
resident	1.4 GB	Resident memory size (+17%)
faults	0/sec	Page faults (none)
connections	20–30	Concurrent connections (increased)
locks (%)	15–25%	Lock wait time (increased)

Table B.12. mongotop measurements (concurrency =1).

Collection	total (ms)	read (ms)	write (ms)	Description
FluorideDB.Sample	480	475	5	Primary read target
SulfideDB-V3.Sample	20	18	2	Secondary access

Table B.13. mongotop measurements (concurrency =10).

Collection	total (ms)	read (ms)	write (ms)	Difference
FluorideDB.Sample	520	510	10	+8.3%
SulfideDB-V3.Sample	25	22	3	+25%

Table B.14. Assumption parameters for performance estimation (measured values).

Parameter	Concurrency 1	Concurrency 10	Change	Impact on performance	Notes
CPI (SELECT)	0.28	0.55	+96%	High	–
L1 miss rate	35%	48%	+37%	Medium	–
SLC miss rate	3%	8%	+167%	High	–
Parallel efficiency	100%	8.4%	-92%	Very high	–
I/O wait time	< 0.5%	< 0.5%	0%	Negligible	–
CPU utilization	8.3%	83.3%	+900%	–	not CPU-limited
Lock wait rate	< 5%	15–25%	+300–400%	High	–

Table B.15. Performance evaluation using a database for all-solid-state battery research.

■ 操作別の推定頻度と実測スループット，推奨並列度

#	操作種別	推定頻度	スループット(ops/sec)	推奨並列度	評価
1	SELECT(Simple)	60%	5,430	1	良好
2	SELECT(Range)	25%	4,330	1	良好
3	SELECT(Complex)	10%	4,384	1	良好
4	INSERT(バッチ)	4%	1,800	10	許容
5	UPDATE	1%	5,623	1	良好

Table B.16. Comprehensive analysis and evaluation of MongoDB benchmarks.

■ 総合分析と評価

#	項目	内容
1	評価軸別分析	i. CRUDにおいてはSELECT操作が15–20倍高速であり，READ-heavy ワークロードに適すると示唆される ii. インデックスにより4–50%の性能向上が確認され，基本インデックスの設定は必須である iii. SELECT操作では並列度の増加が必ずしも有効ではなく，並列化により性能が低下する傾向が確認された iv. INSERT操作では並列化により効果的にスケールし，並列度10において6.4倍の性能向上（並列効率64%）を得た．WRITE操作ではロック競合の影響が相対的に小さい可能性がある
2	ワークロード特性と並列度（同時接続数）の解釈	● **並列度1の利点：** i. 最大スループット：5,430 ops/sec ii. レイテンシ：P50 = 100ms, P99 = 155ms （運用上の支障は小さい） iii. リソース効率：CPU使用率8.3% （省電力であり, 他プロセスとの共存が可能） iv. 安定性：高いキャッシュヒット率を維持 ● **並列度10の問題：** i. スループット低下：4,577 ops/sec (-16%) ii. ドキュメントレベルのロック競合, キャッシュ競合によるメモリアクセス・オーバーヘッド, およびコネクションループ管理のオーバーヘッドが支配的である可能性がある iii. リソース効率：資源使用率の増加にもかかわらず性能が低下 iv. レイテンシは一部改善する一方で，システム全体の効率は悪化
3	システム構成の進化軸（次期計画）	● 分散サーバ構成によるREAD負荷分散 ● プロセッサ更新（M2Pro → M2Max）：L2キャッシュ容量増加による性能改善の検討

Table B.17. Profiling analysis of performance parameters.

■ プロファイリング分析結果（性能パラメータ別）

#	プロファイリング項目	結果	指摘事項
1	CPI分析	● SELECT操作では並列度10において実測CPIが0.28から0.55へ増加(1.96倍)した	● これはWiredTigerのドキュメントレベルロック競合に起因する可能性がある
2	キャッシュミス率	● 並列度10でSLCミス率が3%から8%へ増加(2.7倍)した	● キャッシュ競合により，主記憶(Unified Memory)へアクセスが増加し，レイテンシは改善する一方で，システム全体の効率が低下する可能性がある
3	並列実行効率	● 並列度10において，理論予測値(30.8%)に対し実測値(8.4%)は大幅に低い	● 標準的な理論予測モデルでは不十分であり、MongoDB固有のボトルネック(WiredTiger)を考慮したモデル化が必要である
4	I/O待ち時間	● I/O待ちは，主要な性能ボトルネックではない	● ワーキングセットがWiredTigerキャッシュに収まり，ページフォルトは顕在化していない可能性がある
5	Mongostat/mongotop分析	● 並列度10でロック待ち時間が15–25%増加した	● これは実装データと整合する

付　録 C（Appendix C）

C1. Materials Informatics 連携

Materials Informatics（MI）との連携においては，対象 Collection からデータを直接抽出し，Pandas や SQLite 上で計算・解析を実行する構成も考えられる．これに対し，本開発では研究者の利便性向上を目的として，インタラクティブな GUI を提供した（Fig. C.1–C.3）．さらに，機械学習（ベイズ最適化：PHYSBO と Random Forest を組み合わせた手法）に基づく材料探索支援として，実験候補となる材料組成および合成条件を推薦するアプリケーションを統合している（Fig. C.4–C.8）．これらの MI アプリケーションとの連携により，実験，データ解析，次実験の計画までを一連の流れとして支援でき，実験→解析→次実験までのターンアラウンドタイムの短縮が期待される．

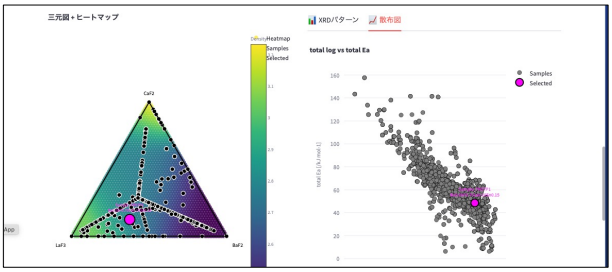


Fig. C.1. Ternary phase diagram (composition mapping and scatter plot of experimental samples).

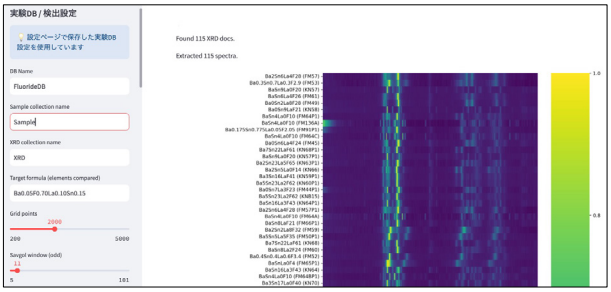


Fig. C.2. XRD pattern analysis/comparison and FWHM-based microstructural analysis.

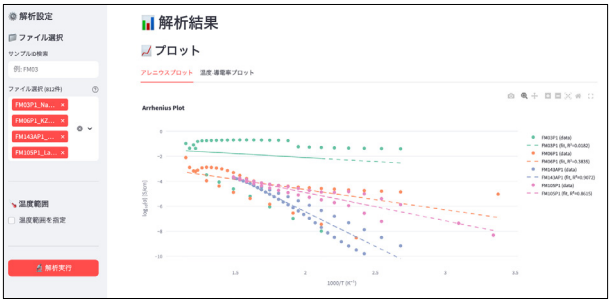


Fig. C.3. Arrhenius analysis: activation energy and conductivity.

Fig. C.4. New material exploration via elemental substitution of a baseline composition.

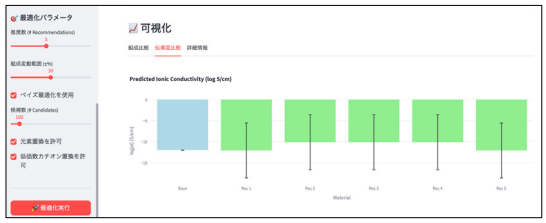


Fig. C.5. Predicted Ionic conductivity.

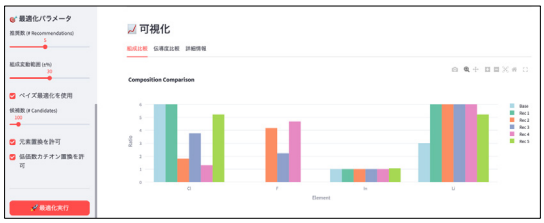


Fig. C.6. Composition comparison.

Fig. C.7. Data-driven inference and recommendation of next candidate compositions and synthesis conditions.

```
✅ API Server: SSB-AI-MCP API Server v0.1.0
✅ Database: connected

🔧 Available tools:
  1. get_sample_count: 実験データベース内のサンプル数を取得
  2. search_xrd_patterns: 化学式によるサンプル検索
  3. get_database_stats: データベース統計情報を取得
  4. get_sample_info: サンプルNoから特定のサンプルの詳細情報を取得
  5. open_xrd_dashboard: XRD分析統合ダッシュボードを開く（Next.js UI）
  6. open_ternary_app: 三元図＋ヒートマップ分析アプリを開く（Streamlit UI）
  7. open_fwhm_analyzer: XRD FWHM（半値全幅）分析アプリを開く（Streamlit UI）
  8. get_ternary_composition_data: 三元図データを取得（3元素系の組成とイオン伝導度データ）
  9. calculate_xrd_fwhm: サンプルNoからXRD FWHMと結晶子サイズを計算
  10. analyze_xrd_peaks: サンプルNoからXRDパターンを分析し、全ピークを検出
  11. recommend_composition: 材料組成の最適化レコメンデーション（ベイズ最適化 or グリッドサーチ）
  12. recommend_material: 材料組成と合成条件の統合最適化レコメンデーション
  13. list_available_databases: 利用可能なMongoDBデータベース一覧を取得
  14. list_collections_in_database: 指定したデータベース内のコレクション一覧を取得
  15. get_collection_info: コレクションの統計情報を取得
  16. query_any_collection: 任意のコレクションに対してMongoDBクエリを実行
  17. search_thinfilm_samples: 薄膜コンビナトリアルデータを検索
  18. get_thinfilm_recipe: 薄膜レシピ情報をレシピIDから取得
  19. search_materials_project: Materials Projectデータベースから材料を検索
  20. get_material_by_id: Materials Project IDから材料情報を取得
  21. get_impedance_data: インピーダンス測定データをサンプル名から取得
  22. analyze_arrhenius_plot: サンプルNoからアレニウスプロット解析を実行
  23. get_synthesis_conditions: サンプルの合成条件を取得
```

Fig. C.8. Key MCP tools used by the recommendation system.