

Çoklu İşlemci Mimarilerinde Kuantum Algoritma Simülasyonlarının Hızlandırılması: Cython, Numba ve Jax ile Optimizasyon Teknikleri

Mehmet Keçeci

ORCID : <https://orcid.org/0000-0001-9937-9839>
mkececi@yaani.com

Received: 03.06.2025

“Article 9 of the series”

Özet/Abstract:

Kuantum algoritmalarının teorik potansiyeli, özellikle karmaşık problemlerin çözümünde devrim niteliğinde olsa da bu algoritmaların pratik geliştirilmesi ve doğrulanması büyük ölçüde klasik bilgisayarlarda yapılan simülasyonlara dayanmaktadır. Ancak, kuantum sistemlerinin Hilbert uzayının üssel büyümesi nedeniyle, artan kübit sayısı ile simülasyonların hesaplama maliyeti hızla artmakta ve bu durum ciddi bir performans darboğazı oluşturmaktadır. Bu zorluğun üstesinden gelmek için, modern çoklu işlemci mimarilerinin sunduğu paralel hesaplama yeteneklerinden faydalanmak kritik bir öneme sahiptir. Bu çalışma (Çoklu İşlemci Mimarilerinde Kuantum Algoritma Simülasyonlarının Hızlandırılması [Unpublished pre-doctoral IX. Report]. Gebze Technical University, Kocaeli, Türkiye [467, 478–480]), Python programlama dili kullanılarak geliştirilen kuantum algoritma simülasyonlarının performansını artırmak amacıyla Cython, Numba ve Jax gibi ileri düzey optimizasyon araçlarının kullanımını ve bu araçların çoklu işlemci ortamlarındaki etkinliklerini incelemektedir. Cython, Python kodunu C veya C++’a çevirerek statik tipleme ve derleme avantajları sunar, bu sayede Python’ın yorumlama yükünü ortadan kaldırır ve GIL kısıtlamasını belirli koşullar altında aşarak gerçek iş parçacığı düzeyinde paralellik sağlar. Numba, JIT derleyicisi ile Python fonksiyonlarını, özellikle NumPy dizileri üzerinde çalışan sayısal hesaplama ağırlıklı döngüleri, çalışma zamanında makine koduna çevirerek önemli hızlanmalar elde eder; ayrıca “@jit(parallel=True)” gibi direktiflerle otomatik paralelleştirme yetenekleri sunar. Jax ise, otomatik farklılaştırma ve XLA derleyicisi ile optimizasyonun yanı sıra, “pmap” fonksiyonu aracılığıyla veri paralelliği modelini destekleyerek işlemleri birden fazla CPU çekirdeği veya GPU/TPU gibi hızlandırıcılara dağıtılmasını kolaylaştırır ve “vmap” ile otomatik vektörleştirmeyi mümkün kılar. Bu optimizasyon tekniklerinin entegrasyonu ve etkin kullanımı, kuantum algoritma simülasyonlarının yürütme sürelerini önemli ölçüde azaltabilir. Çalışma, bu araçların bireysel ve birleşik etkilerini, özellikle kuantum hata düzeltme kodları veya varyasyonel kuantum algoritmaları gibi hesaplama açısından yoğun görevlerin simülasyonunda ele almaktadır. Elde edilen performans kazanımları, daha büyük ve daha karmaşık kuantum sistemlerinin klasik kaynaklarla incelenmesine olanak tanıyarak, kuantum bilişim alanındaki araştırmaların ilerlemesine katkıda bulunacaktır. Sonuç olarak, Cython, Numba ve Jax’ın sunduğu derleme ve paralelleştirme stratejileri, kuantum simülasyonlarının çoklu işlemci mimarilerinde verimli bir şekilde çalıştırılması için güçlü ve esnek çözümler sunmaktadır.

Anahtar Kelimeler/ Keywords:

Kuantum Algoritma Simülasyonu, Çoklu İşlemci Mimarileri, Performans Optimizasyonu, Paralel Hesaplama, Cython, Numba, Jax, Python, Yüksek Başarımli Hesaplama, Kuantum Bilişim.

Note: Citations and numbering are in continuation of the previous articles.

Accelerating Quantum Algorithm Simulations in Multi-Processor Architectures: Optimisation Techniques with Cython, Numba, and Jax

Mehmet Keçeci

ORCID : <https://orcid.org/0000-0001-9937-9839>
mkececi@yaani.com

Received: 07.10.2025

“Article 9 of the series”

Abstract:

While the theoretical potential of quantum algorithms is revolutionary, particularly for solving complex problems, their practical development and validation rely heavily on simulations performed on classical computers. However, due to the exponential growth of the Hilbert space of quantum systems, the computational cost of simulations increases rapidly with the number of qubits, creating a significant performance bottleneck. To overcome this challenge, leveraging the parallel computing capabilities offered by modern multi-processor architectures is of critical importance. This study (Accelerating Quantum Algorithm Simulations in Multi-Processor Architectures [Unpublished pre-doctoral IX. Report]. Gebze Technical University, Kocaeli, Türkiye [467, 478–480]) investigates the use of advanced optimisation tools—namely Cython, Numba, and Jax—to enhance the performance of quantum algorithm simulations developed using the Python programming language, and examines their efficacy in multi-processor environments. Cython translates Python code into C or C++, offering the advantages of static typing and compilation. This process eliminates Python’s interpretation overhead and, under specific conditions, bypasses the Global Interpreter Lock (GIL) constraint, enabling genuine thread-level parallelism. Numba employs a Just-In-Time (JIT) compiler to translate Python functions, particularly those with numerically intensive loops operating on NumPy arrays, into machine code at runtime, achieving significant speedups. It also provides automatic parallelisation capabilities through directives such as “@njit(parallel=True)”. Jax, through its automatic differentiation and the XLA compiler, offers optimisation and supports a data parallelism model via its “pmap” function. This facilitates the distribution of operations across multiple CPU cores or accelerators like GPUs/TPUs, while “vmap” enables automatic vectorisation. The integration and effective use of these optimisation techniques can substantially reduce the execution times of quantum algorithm simulations. This work assesses the individual and combined impact of these tools in simulating computationally intensive tasks, such as quantum error correction codes and variational quantum algorithms. The resultant performance gains will facilitate the study of larger and more complex quantum systems using classical resources, thereby contributing to the advancement of research in quantum computing. In conclusion, the compilation and parallelisation strategies provided by Cython, Numba, and Jax offer powerful and flexible solutions for running quantum simulations efficiently on multi-processor architectures.

Keywords:

Quantum Algorithm Simulation, Multi-Processor Architectures, Performance Optimisation, Parallel Computing, Cython, Numba, Jax, Python, High-Performance Computing, Quantum Computing.

Note: Citations and numbering are in continuation of the previous articles.

I. Klasik Yüksek Başarımlı Hesaplama ve Optimizasyon Araçlarının Kuantum Dünyasına Köprüsü: Simülasyonlarda Performansın Kilidini Açmak

Yirminci yüzyılın başlarında filizlenen kuantum mekaniği, doğanın mikroskobik düzeydeki işleyişine dair anlayışımızı kökten değiştirmiş ve teknolojik devrimlerin önünü açmıştır. Günümüzde, kuantum bilgisayarlar, kuantum iletişim ve kuantum sensörler gibi yenilikçi teknolojiler, bu temel bilimin somut uygulamaları olarak karşımıza çıkmaktadır. Kuantum bilgisayarlar, özellikle belirli problem türlerinde (örneğin, büyük sayıların asal çarpanlarına ayrılması, karmaşık moleküllerin simülasyonu, optimizasyon problemleri) klasik bilgisayarların yeteneklerini aşan bir hesaplama gücü vaat etmektedir. Ancak, fiziksel olarak büyük, kararlı ve hatasız kuantum bilgisayarların geliştirilmesi, eşevresizlik (decoherence), kübitler arası etkileşimlerin kontrolü ve hata düzeltme gibi bir dizi zorlu teknik engelle karşı karşıyadır. Bu engeller, kuantum donanımlarının gelişimini yavaşlatırken, kuantum algoritmalarının ve kuantum sistemlerinin davranışlarının anlaşılması ve doğrulanması için klasik bilgisayarlar üzerinde yapılan simülasyonlara olan ihtiyacı artırmaktadır.

Kuantum sistemlerinin simülasyonu, özünde, bu sistemleri tanımlayan Schrödinger denkleminin veya eşdeğer formülasyonların sayısal olarak çözülmesini içerir. Ne var ki, kuantum mekaniğinin temel prensiplerinden biri olan süperpozisyon, bir kuantum sisteminin durumunu tanımlamak için gereken parametre sayısının (Hilbert uzayının boyutu) sistemdeki kübit sayısı ile üssel olarak artmasına neden olur. Örneğin, N kübitlik bir sistemi tam olarak tanımlamak için 2^N karmaşık sayıya ihtiyaç duyulur. Bu “boyutsallık laneti” (curse of dimensionality), mütevazı sayıda kübite sahip sistemlerin bile simülasyonunu klasik bilgisayarlar için son derece hesaplama-yoğun ve bellek-yoğun bir görev haline getirir. Dolayısıyla, birkaç on kübitten fazla sistemi tam olarak simüle etmek, günümüzün en güçlü süper bilgisayarları için bile bir meydan okumadır.

Bu zorlukların üstesinden gelmek ve kuantum araştırmalarını ilerletmek için, klasik hesaplama paradigmalarının ve yazılım araçlarının sınırlarını zorlamak gerekmektedir. İşte bu noktada, Yüksek Başarımlı Hesaplama (High-Performance Computing - HPC) altyapıları ve modern yazılım optimizasyon teknikleri devreye girer. HPC, çok sayıda işlemci çekirdeğini (CPU’lar), grafik işlem birimlerini (GPU’lar) ve özel hızlandırıcıları barındıran paralel mimariler kullanarak devasa hesaplama görevlerinin üstesinden gelmeyi amaçlar. Bu mimariler, büyük ölçekli kuantum simülasyonları için gerekli olan ham hesaplama

gücünü ve bellek kapasitesini sunar. Paralel programlama modelleri (örneğin, MPI, OpenMP) ve dağıtık bellek sistemleri, simülasyon yükünün birden fazla işlem birimine verimli bir şekilde dağıtılmasını sağlar.

Ancak, ham donanım gücü tek başına yeterli değildir. Kuantum simülasyon kodlarının bu güçlü donanımlardan etkin bir şekilde yararlanabilmesi için yazılım düzeyinde ciddi optimizasyonlara ihtiyaç vardır. Bilimsel hesaplama alanında yaygın olarak kullanılan Python gibi yüksek seviyeli diller, geliştirme kolaylığı ve zengin ekosistemleri nedeniyle popüler olsa da yorumlanmış doğaları ve Global Interpreter Lock (GIL) gibi kısıtlamaları nedeniyle saf performans açısından genellikle C, C++ veya Fortran gibi derlenmiş dillerin gerisinde kalırlar. Kuantum simülasyonlarında karşılaşılan karmaşık matematiksel işlemler ve yoğun döngüler, bu performans farkını daha da belirgin hale getirir.

Bu açığı kapatmak ve Python'un esnekliği ile derlenmiş dillerin performansını birleştirmek amacıyla Cython, Numba ve Jax gibi araçlar geliştirilmiştir. Bu araçlar, kuantum hesaplama topluluğu için paha biçilmez varlıklar haline gelmiştir:

1. **Cython:** Python kodunun C veya C++ koduna çevrilmesini sağlayan bir derleyicidir. Geliştiricilere, Python sözdizimine çok yakın bir dille yazarak, kritik performans gerektiren kod bölümlerini statik tiplendirme ile optimize etme ve doğrudan C kütüphaneleriyle etkileşim kurma imkânı sunar. Cython, GIL'i serbest bırakma yeteneği sayesinde, CPU-yoğun görevlerde gerçek iş parçacığı düzeyinde paralellik (thread-level parallelism) sağlayarak çok çekirdekli işlemcilerden daha iyi faydalanılmasına olanak tanır. Kuantum simülasyonlarında, zaman gelişimi operatörlerinin uygulanması, Hamiltonian matrislerinin oluşturulması veya karmaşık lineer cebir işlemlerinin hızlandırılması gibi görevlerde etkin bir şekilde kullanılabilir.
2. **Numba:** Python fonksiyonlarını ve özellikle NumPy dizileri üzerinde çalışan sayısal döngüleri Just-In-Time (JIT) derleme tekniği ile çalışma zamanında optimize edilmiş makine koduna dönüştüren bir derleyicidir. Numba, @jit veya @njit gibi basit dekoratörler aracılığıyla kullanılır ve geliştiricinin kodunda büyük değişiklikler yapmasını gerektirmez. Özellikle yoğun sayısal hesaplamalar içeren kuantum algoritmalarının (örneğin, kuantum Fourier dönüşümü, kuantum faz kestirimi) veya kuantum sistemlerinin dinamiklerini simüle eden adi diferansiyel denklem çözücülerinin hızlandırılmasında büyük fayda sağlar. Numba ayrıca, CUDA (GPU'lar için) ve ROCm (AMD GPU'lar için – gelişmekte olan destek) aracılığıyla GPU hızlandırmasını

ve @vectorize, @guvectorize ile SIMD (Single Instruction, Multiple Data) operasyonlarını destekler. @njit(parallel=True) seçeneği ile bazı döngülerin otomatik olarak paralelleştirilmesini de mümkün kılar.

3. **Jax:** Temelde NumPy benzeri bir API üzerine kurulu, otomatik farklılaştırma (automatic differentiation) ve XLA (Accelerated Linear Algebra) derleyicisi ile yüksek performanslı sayısal hesaplama için tasarlanmış bir Python kütüphanesidir. Otomatik farklılaştırma özelliği, özellikle varyasyonel kuantum algoritmaları (VQA'lar) ve kuantum makine öğrenmesi modelleri gibi gradyan tabanlı optimizasyon gerektiren alanlarda kritik öneme sahiptir. Jax, jit dekoratörü ile fonksiyonları XLA aracılığıyla derleyerek GPU'lar ve TPU'lar (Tensor Processing Units) üzerinde verimli bir şekilde çalıştırabilir. pmap (parallel map) fonksiyonu, veri paralellliği sağlayarak işlemleri birden fazla cihaza (örneğin, çoklu GPU) kolayca dağıtmayı mümkün kılar. Vmap (vectorizing map) otomatik olarak toplu işlemler (batching) için kodun vektörleştirilmesini sağlar. Bu yetenekler, büyük ölçekli kuantum devre simülasyonlarında ve kuantum sistemlerinin parametrik optimizasyonunda Jax'ı güçlü bir araç haline getirir.

Bu araçların yanı sıra, Python'un standart kütüphanesinde bulunan multiprocessing ve concurrent.futures gibi modüller de görevlerin birden fazla işlemci çekirdeğine dağıtılması yoluyla süreç tabanlı paralellik (process-based parallelism) sağlayarak GIL kısıtlamasının etrafından dolaşır. Bu, özellikle birbirinden bağımsız çok sayıda simülasyon çalışmasının (örneğin, parametre taramaları, Monte Carlo simülasyonları) paralel olarak yürütülmesinde kullanışlıdır.

Kuantum hesaplamalarında bu sistem ve yazılımların önemi birkaç katmanlıdır. İlk olarak, **erişilebilirlik ve geliştirme hızı** sunarlar. Python'un öğrenme kolaylığı ve geniş kütüphane desteği, farklı disiplinlerden araştırmacıların kuantum algoritmaları geliştirmesini ve test etmesini kolaylaştırır. Cython, Numba ve Jax gibi araçlar, bu yüksek seviyeli ortamın performansını önemli ölçüde artırarak, araştırmacıların fikirlerini daha hızlı prototiplemesine ve daha büyük sistemleri keşfetmesine olanak tanır. İkinci olarak, **mevcut HPC altyapılarından en iyi şekilde yararlanmayı** sağlarlar. Bu araçlar, karmaşık paralel programlama detaylarını soyutlayarak veya basitleştirerek, kuantum simülasyon kodlarının modern çok çekirdekli CPU'lar ve GPU'lar üzerinde verimli bir şekilde çalıştırılmasına yardımcı olur. Bu, daha önce pratik olmayan ölçeklerdeki simülasyonların gerçekleştirilmesini mümkün kılar. Üçüncü olarak, **yeni kuantum algoritmalarının ve donanım tasarımlarının keşfini ve doğrulanmasını** hızlandırırlar.

Simülasyonlar, yeni kuantum algoritmalarının davranışlarını anlamak, performanslarını değerlendirmek ve potansiyel hata kaynaklarını belirlemek için vazgeçilmezdir. Optimize edilmiş simülasyonlar, bu keşif sürecini önemli ölçüde kısaltabilir. Ayrıca, farklı kübit tasarımlarının veya hata düzeltme kodlarının etkinliğini test etmek için de kritik bir rol oynarlar.

Sonuç olarak, kuantum bilgisayarların tam potansiyeline ulaşması zaman alacak olsa da, klasik bilgisayarlar üzerinde çalışan gelişmiş simülasyonlar, bu heyecan verici alandaki ilerlemenin temel taşlarından biri olmaya devam edecektir. Paralel mimariler, HPC sistemleri ve Cython, Numba, Jax gibi yazılım optimizasyon araçları, kuantum dünyasının karmaşıklıklarını klasik kaynaklarla modellemek için güçlü bir köprü görevi görerek, kuantum simülasyonlarının sınırlarını genişletmekte ve kuantum bilişimin geleceğini şekillendirmede hayati bir rol oynamaktadır. Bu araçların bütünleşmiş ve akılcı kullanımı, hesaplama bilimcilerine ve kuantum araştırmacılarına, daha önce erişilemez olan problem alanlarına girme ve kuantumun gizemlerini çözme yolunda önemli bir avantaj sağlamaktadır.

II. Simülasyon Kodlarında Performans Optimizasyonu ve Kuantum Hesaplamalarda Uygulanabilirliği

Simülasyon (benzetim) kodlarını Python, Xeus Python (XPython) ve SageMath (Linux) platformlarının her üçünde de sorunsuz çalışmaktadır. SageMath (sürüm 10.1-10.6) kullanımı, oldukça kararlı ve yüksek performanslı sonuçlar vermiştir. Örneğin, optimize edilmemiş bir Python kodu ile π sayısının 20.000 basamağını hesaplamak 33 saat sürerken ve sonuç alınamazken, SageMath'in kendi içsel fonksiyonları kullanılarak π sayısının 1 milyar basamağı (1 GB metin çıktısı) yaklaşık 2,5 saatte hesaplanabilmiştir.

15 hata (2024) içeren (17 hata, 2025) simülasyon kodları, öncelikle kendi içlerinde optimize edilerek (seri optimizasyonlarla) yaklaşık 294.629 kattan 320 kata varan bir hızlanma göstermiştir. Ardından, bu optimize edilmiş kodlara paralel işleme (çoklu işlemci) desteği eklenerek toplamda yaklaşık 435.570 kat hızlanma sağlanmıştır. Bu süreçte aşağıdaki modüllerden yararlanılmıştır:

Yapıların Detaylı Açıklamaları:

1. **Cython:** Cython, Python programlama dilinin C ve C++ dilleriyle harmanlandığı bir üst kümesidir. Temel amacı, Python kodunun performansını C seviyesine yaklaştırmaktır. Bunu, Python kodunu C veya C++ koduna çevirerek (transpilasyon) ve ardından bu C/C++ kodunu bir C derleyicisi (örn: GCC, Clang) aracılığıyla makine koduna derleyerek yapar. Cython, geliştiricilere Python'un esnekliğini ve C'nin ham hızını bir arada kullanma imkânı tanır. Kritik kod bölümlerinde statik tip tanımlamaları yaparak Python'un dinamik tip denetiminden kaynaklanan ek yükü ortadan kaldırabilir. En önemli özelliklerinden biri, nogil direktifi ile Global Interpreter Lock (GIL) kısıtlamasını belirli bloklarda kaldırabilmesidir. Bu, CPU-yoğun görevlerde gerçek iş parçacığı düzeyinde paralellik (thread-level parallelism) sağlayarak çok çekirdekli işlemcilerden tam anlamıyla faydalanılmasına olanak tanır. Ayrıca C/C++ kütüphaneleriyle doğrudan ve verimli bir şekilde arayüz oluşturmak için de idealdir.
2. **Numba:** Numba, Python fonksiyonlarını, özellikle de NumPy dizileri üzerinde çalışan sayısal hesaplama ağırlıklı döngüleri, çalışma zamanında (Just-In-Time - JIT) optimize edilmiş makine koduna dönüştüren bir derleyicidir. Kullanımı oldukça basittir; genellikle hızlandırılmak istenen fonksiyonun üzerine @jit veya @njit (nopython modu için, en iyi performansı sunar) gibi bir dekoratör eklemek yeterlidir. Numba, LLVM derleyici altyapısını kullanarak Python bytecode'unu analiz eder ve yüksek performanslı makine kodu üretir. Özellikle bilimsel hesaplama ve veri analizi alanlarında, karmaşık döngülerin ve matematiksel işlemlerin bulunduğu yerlerde saf Python'a göre önemli hızlanmalar sağlar. CUDA ve ROCm (gelişmekte olan destek) aracılığıyla GPU'lar üzerinde kod çalıştırma yeteneği sunar. Ayrıca, @vectorize ve @guvectorize dekoratörleri ile evrensel fonksiyonlar (ufuncs) oluşturulabilir ve @njit(parallel=True) seçeneği ile belirli döngü yapıları otomatik olarak paralelleştirilebilir.
3. **Jax:** Jax, Google tarafından geliştirilen ve NumPy benzeri bir API üzerine kurulu, yüksek performanslı sayısal hesaplama ve makine öğrenmesi için tasarlanmış bir Python kütüphanesidir. En ayırt edici özellikleri otomatik farklılaştırma (automatic differentiation - grad fonksiyonu), XLA (Accelerated Linear Algebra) derleyicisi ile optimizasyon ve GPU/TPU gibi hızlandırıcılarda verimli çalışabilmesidir. Jax, fonksiyonel programlama paradigmasını benimser (fonksiyonlar saf olmalıdır). jit dekoratörü ile fonksiyonları XLA aracılığıyla derleyerek önemli performans kazanımları sağlar. vmap (vectorizing map) fonksiyonu, işlemleri otomatik olarak vektörleştirerek toplu (batch) işlemler için kolaylık sunarken, pmap (parallel map) fonksiyonu ise veri paralellliği modelini kullanarak işlemleri birden fazla CPU çekirdeğine veya birden fazla GPU/TPU cihazına

dağıtmayı mümkün kılar. Bu özellikleri, özellikle derin öğrenme modellerinin eğitimi ve karmaşık bilimsel simülasyonlar için Jax'ı güçlü bir araç haline getirir.

4. **concurrent.futures.ThreadPoolExecutor:** Python'un standart concurrent.futures modülünün bir parçası olan ThreadPoolExecutor, görevlerin eşzamanlı (concurrent) olarak bir iş parçacığı havuzunda (thread pool) çalıştırılmasını kolaylaştıran üst düzey bir arayüz sunar. İş parçacıkları, aynı süreç (process) içinde çalıştıkları için bellek alanını paylaşırlar, bu da veri paylaşımını kolaylaştırır ancak senkronizasyon sorunlarına dikkat edilmesini gerektirir. ThreadPoolExecutor, özellikle G/Ç-yoğun (I/O-bound) görevler için uygundur. Bu tür görevlerde (örneğin, ağ istekleri, dosya okuma/yazma), iş parçacıkları G/Ç işleminin tamamlanmasını beklerken diğer iş parçacıkları çalışmaya devam edebilir, böylece bekleme süreleri etkin bir şekilde kullanılır. Ancak, Python'un Global Interpreter Lock (GIL) kısıtlaması nedeniyle, CPU-yoğun (CPU-bound) görevlerde ThreadPoolExecutor genellikle gerçek bir performans artışı sağlamaz, çünkü aynı anda sadece bir Python iş parçacığı Python bytecode'unu çalıştırabilir.
5. **concurrent.futures.ProcessPoolExecutor:** Yine concurrent.futures modülünün bir parçası olan ProcessPoolExecutor, görevlerin paralel olarak ayrı süreçlerde (process pool) çalıştırılmasını sağlar. Her süreç kendi Python yorumlayıcısına ve bellek alanına sahip olduğu için, ProcessPoolExecutor GIL kısıtlamasını aşar. Bu da onu CPU-yoğun görevler için ideal bir çözüm yapar, çünkü görevler farklı işlemci çekirdeklerinde gerçekten paralel olarak çalışabilir. Görevler ve sonuçlar süreçler arasında serileştirilerek (pickling) aktarılır, bu da bazı ek yüklerle neden olabilir, özellikle büyük veri yapıları için. ThreadPoolExecutora benzer bir arayüz sunar, bu da aralarında geçiş yapmayı kolaylaştırır.
6. **multiprocessing:** Python'un standart kütüphanesinde yer alan multiprocessing modülü, süreç tabanlı paralellik oluşturmak için daha düşük seviyeli ve daha kapsamlı araçlar sunar. ProcessPoolExecutor gibi, GIL'i baypas ederek CPU-yoğun görevlerde gerçek paralellik sağlar. Process, Pool, Queue, Pipe, Lock, Semaphore gibi çeşitli sınıflar ve mekanizmalar içerir. Bu, süreçler arasında karmaşık iletişim ve senkronizasyon senaryolarının uygulanmasına olanak tanır. ProcessPoolExecutor'a göre daha fazla esneklik ve kontrol sunar, ancak kullanımı biraz daha karmaşık olabilir. Özellikle birbirinden büyük ölçüde bağımsız olan ve önemli hesaplama gücü gerektiren görevlerin paralelleştirilmesi için çok etkilidir.

Bu modüllerden Cython, Numba ve Jax, kodu C/C++ seviyesine derleyerek hızlandırırken; concurrent.futures ve multiprocessing modülleri ise paralel işlem yetenekleri sunmaktadır.

Cython, Numba ve Jax, çeşitli paralelleştirme teknikleri sunarak paralel işlemlerde kritik rol oynar. Bu araçların concurrent.futures ve multiprocessing modülleriyle entegrasyonu, paralel hesaplamaların verimliliğini daha da artırabilir; nitekim bu çalışmada da verim artışı gözlemlenmiştir.

Bu optimizasyon çabalarının temel amacı, kuantum bilgisayarlardaki eşevrelik (coherence) süresi kısıtlamasıdır. Kübit sayısı arttıkça hata oranları da doğal olarak artmaktadır. Eşevrelik süresi, bir kübitin süperpozisyon ve dolanıklık gibi kuantum özelliklerini koruyabildiği zaman aralığını ifade eder. Bu süre, kuantum bilgi işleminin verimliliği ve doğruluğu için kritik öneme sahiptir, zira kuantum hesaplamalarının gerçekleştirilebilmesi için kübitlerin bu durumları kararlı bir şekilde sürdürebilmesi gerekir. Eşevrelik süresi aşıldığında, kübitler klasik davranış sergilemeye başlar ve kuantum hesaplamaları yapılamaz hale gelir. Bu süre; çevresel gürültü, malzeme kusurları, termal dalgalanmalar ve diğer dış etkileşimlerden kaynaklanan eşevresizlik (decoherence) süreçleri nedeniyle sınırlıdır ve kuantum bilgisayarının türüne ve teknolojisine bağlı olarak mikrosaniyelerden saniyelere kadar değişebilir. Bu nedenle, düşük gürültülü veya gürültüsüz kuantum bilgisayarların geliştirilmesi büyük önem taşımaktadır. Daha uzun eşevrelik süreleri, daha karmaşık ve uzun hesaplamaların yapılmasını mümkün kılar.

Majorana ve Weyl fermiyonlarının kullanım amaçlarından biri, eşevrelik sürelerini uzatarak daha güçlü ve güvenilir kuantum bilgisayarlar geliştirmektir. Bir diğer yaklaşım ise, bu simülasyon çalışmasında uygulandığı gibi, eşevrelik süresi uzatılamıyorsa, işlemleri çoklu işlem birimlerine (çoklu CPU & GPU, süper bilgisayarlar) bölerek hesaplamayı eşevrelik süresi içinde tamamlamaktır. Ancak mevcut durumda genellikle tek bir QPU (Kuantum İşlem Birimi) kullanıldığından, işlemlerin bir kısmı klasik bilgisayarlarda çözülmektedir.

Kuantum işlemlerini şu şekilde sınıflandırabiliriz:

1. **Tamamen Kuantum Özelliklerine Sahip Olan İşlemler:** Kuantum süperpozisyonu ve dolanıklık gibi prensiplere dayanır ve klasik bilgisayarlar tarafından verimli bir şekilde simüle edilemez.
2. **Klasik İşlemlere Çok Yakın Sonuç Veren Kuantum İşlemleri:** Kuantum bilgisayarların sonuçlarının klasik bilgisayar sonuçlarına yakın olduğu, ancak potansiyel kuantum hızlanması veya başka avantajlar sunabileceği işlemlerdir.

3. **Klasik İşlemlerle Aynı Sonucu Veren Kuantum İşlemleri:** Kuantum bilgisayarlar tarafından gerçekleştirilse de sonuçları klasik bilgisayarlarınkıyla aynıdır; genellikle kuantum sistemlerinin klasik davranışları taklit ettiği durumlardır.

Bu çalışmada temel amacım, kuantum hata düzeltme işlemlerini paralel hesaplama teknikleri kullanarak gerçekleştirip, eşevrelilik süresi kısıtlamasının üstesinden gelerek sonuca ulaşmaktır. Bu tür hibrit yaklaşımlar (kuantum bilgisayarı + klasik işlemciler/süper bilgisayar desteği) halihazırda kullanılmaktadır. Bu yaklaşımı uygulayarak hem simülasyon kodumuzun çalışabilirliğini test ettim hem de önemli bir performans artışı elde ettim. Bu çalışmanın önemli bir sonucu, yüksek kübit sayılı kuantum sistemlerinin analizinde, paralel klasik hesaplama kaynaklarından yararlanılarak eşevrelilik süresi kısıtlamalarının aşılabileceğidir.

Senaryo	İşlem Süresi	Bağıl Hızlanma	Özyineleme Sayısı (Yaklaşık)	Kullanılan Araçlar/Diller (Optimize Durum)
6561 Kübit, 9 Hata Optimize Edilmemiş Python 3.11	113.972,37 sn (≈31,66 saat)	1x (Referans)	7,7 x 10 ⁹	Python
6561 Kübit, 15 Hata Optimize Edilmiş ve Paralleleştirilmiş Kod	261,66 sn (≈4,36 dakika)	≈435.570x	6	Python (3.8-3.13), Xeus Python, SageMath, Cython, Numba, Jax, vb.
Ulaşılan son değerler: Grafiksiz, 17 hata çözümü: 25.000.000 kübit 2 Boyut grafikli, 17 hata çözümü: 250.000 kübit 3 Boyutlu, 17 hata çözümü: 125.000 kübit Düzlem: Planar, Toric, Rotated, Cylinder, Cubic (Düzlem, Torik, Döndürülmüş, Silindir, Kübik)				

Tablo 21: Kodlama Dili ve Optimizasyon Seviyesine Göre Performans Karşılaştırması (2B Torik Kod, MWPM Algoritması)

Not: Hızlanma, 15 hata durumu için optimize edilmemiş Python implementasyonuna kıyasla hesaplanmıştır (optimize edilmemiş durumun süresi tabloda doğrudan verilmemiştir ancak 9 hata durumu referans alınarak büyüklük kestirilebilir). Özyineleme sayısı, algoritmanın karmaşıklığını ve çözüme

ulaşmak için gereken adım sayısını gösterir; optimizasyonlarla ciddi oranda azaltılmıştır. Testler Ubuntu 24,04 ve Windows 10 üzerinde yapılmıştır.

Ayrıca, simülasyon kodu NumPy 2.0.0-2.2.6, Matplotlib 3.9.0-3.10.3, SciPy 1.14.0-1.15.3, Qiskit 1.1.1-2.0.2 ve Rustworkx 0.15.1-0.16.0, NetworkX, igraph, Networkit, Graphillion (Windows, Linux), graph-tool (Linux) gibi güncel modüllerle de sorunsuz bir şekilde çalışmaktadır. Qiskit modülü, simülasyonda uygulanan 17 hatadan sadece bir tanesinin çözümünde kullanılmış olup, Qiskit'in esnek yapısı sayesinde farklı hata modellerinin entegrasyonu da mümkündür. Rustworkx modülü ise Rust ile yazılmış Blossom algoritması veya diğer graf algoritmalarının doğrudan kullanılabilmesini sağlayarak kodun yeteneklerini artırmakta ve önemli bir avantaj sunmaktadır.

Majorana ve Weyl kübitleri gibi topolojik kübitler, teorik olarak daha uzun eşevrelik süreleri vaat etse de bu çalışmada sunulan paralel işleme gibi alternatif yöntemler, mevcut ve yakın gelecekteki teknolojiler için değerli yaklaşımlar olarak değerlendirilmektedir. Weyl fermiyonları, yalnızca kuantum bilgisayarlar için değil, aynı zamanda ileri düzey malzeme bilimi ve elektronik uygulamalarında da yeni ufuklar açabilecek potansiyele sahiptir ve topolojik özellikleri nedeniyle topolojik kübitler için önemli bir adaydır.

III. Kuantum Hata Düzeltmede Metriklerin Geleceği: Akıllı Kod Çözücülerden Donanım-Metrik Eş Tasarımına

Özellik	Python	Cython	Xeus-Python (XPython)	SageMath
Temel Tanım	Yüksek seviyeli, yorumlanan, genel amaçlı programlama dili.	Python kodunu C/C++'a derleyerek performansı artırır; Python ile tam uyumlu.	Jupyter kernel protokolüne dayalı, C++ ile yazılmış interaktif Python çekirdeği.	Python temelli matematiksel yazılım sistemi; sembolik ve sayısal hesaplamalar için zengin ortam.
Performans	Yavaş olabilir (özellikle CPU-yoğun işlemlerde).	C/C++ seviyesine yakın hız; “nogil”	Standart Python (CPython) hızında; Jupyter	Matematiksel işlemler için yüksek performans

		ile paralel işlem yapılabilir.	entegrasyonu için optimize.	(C/Fortran kütüphaneleri sayesinde).
Derleme	Yorumlanır (bytecode'a derlenir).	Python → C/C++ → makine koduna dönüşüm.	Yorumlanır (standart Python gibi).	Python kodu yorumlanır; ancak içsel fonksiyonlar derlenmiş kodlara dayanır.
GIL Yönetimi	GIL var (3.13 öncesi); gerçek iş parçacığı paralelliği yok.	“nogil” bloklarıyla GIL serbest bırakılabilir.	Standart Python gibi GIL'e tabidir.	Python GIL'e bağlı, ancak C/C++ eklentileriyle paralel çalışabilir.
Kullanım Amacı	Web geliştirme, veri bilimi, yapay zeka, betikleme.	Performans kritik Python kodları ve C/C++ entegrasyonu.	Jupyter notebook/labs ile interaktif çalışma.	Sembolik hesaplama, cebir, sayılar teorisi, kriptografi, grafikler.
Sözdizimi	Standart Python sözdizimi.	Python'ın üst kümesi; statik tipler eklenir.	Standart Python.	Python + özel matematiksel ifadeler ve nesneler.
Entegrasyon	Geniş Python paketleri (NumPy, Pandas vs.).	C/C++ ile doğrudan entegrasyon; NumPy desteği güçlü.	Jupyter notebook, lab, widget entegrasyonu mükemmel.	NumPy, SciPy, GAP, PARI/GP gibi araçlarla entegre.
Paralel İşleme	“multiprocessing”, “threading”, “asyncio”.	“prange”, OpenMP ile paralelleştirme desteklenir.	Python'daki modüllerle sınırlıdır.	Python paralel modülleri + bazı dahili algoritmalar.
Öğrenme Eğrisi	Kolay öğrenilebilir.	Python bilmek yeterli ama GIL,	Jupyter bilgisi yeterlidir.	Python bilgisi + matematiksel

		tip tanımlama gibi konular gerekir.		arayüzlerin öğrenilmesi gerekebilir.
Avantajları	Okunabilirlik, hızlı geliştirme, büyük topluluk.	C düzeyinde performans, kolay C entegrasyonu, Python esnekliği.	Jupyter ile harika uyum, interaktif deneyim.	Zengin matematiksel araç seti, tek pencerede tüm çözümler.
Dezavantajları	Düşük performans, GIL (eski sürümlerde).	Derleme adımı, daha karmaşık yapı.	Standart Python hızında; ağır uygulamalar için yetersiz kalabilir.	Büyük kurulum, uyumluluk sorunları olabilir.

Tablo 22: Karşılaştırma Tablosu

Özellik	Cython	Numba	Jax	“ThreadPoolExecutor”	“ProcessPoolExecutor”	“multiprocessing”
Temel Prensipler	Python’u C/C++’a çevirip derleme (AOT).	Python fonksiyonlarını JIT derleme ile makine koduna çevirme.	NumPy benzeri API, JIT derleme (XLA), otomatik türev alma.	Görevleri iş parçacığı havuzunda eşzamanlı çalıştırma.	Görevleri süreç havuzunda paralel çalıştırma.	Süreç tabanlı paralellik için kapsamlı araçlar.
Derleme Türü	Ön Derleme (AOT - Ahead-of-Time)	Anında Derleme (JIT - Just-In-Time)	Anında Derleme (JIT - XLA aracılığıyla)	Yok (Python yorumlanır)	Yok (Python yorumlanır)	Yok (Python yorumlanır)
GIL Yönetimi	“nogil” ile serbest	“nopython”	Python kısmı	GIL’e tabidir, CPU-yoğun	GIL’i baypas eder (her süreç)	GIL’i baypas eder.

	bırakılabilir.	modunda belirli işlemlere serbest bırakılabilir.	GIL'e tabi, XLA derlenmiş kerneller etkilenmez.	görevlerde paralellik kısıtlıdır.	kendi yorumlayıcısına sahip).	
Paralellik Türü	İş parçacığı (OpenMP ile), Görev	İş parçacığı (otomatik döngü), SIMD, Veri (GPU)	Veri ("pmap"), Görev (fonksiyon el kompozisyon), Vektörleştirme ("vmap")	Eşzamanlılık (iş parçacığı)	Paralellik (süreç)	Paralellik (süreç)
Ana Kullanım Alanı	Python hızlandırma, C/C++ entegrasyonu.	Sayısal Python kodunu (öz. NumPy döngüleri) hızlandırma, bilimsel hesaplamalar.	Makine öğrenmesi, sayısal optimizasyon, GPU/TPU üzerinde yüksek performanslı hesaplama.	G/Ç-yoğun görevler (ağ, dosya işlemleri).	CPU-yoğun görevler.	CPU-yoğun görevler, karmaşık paralel iş akışları.
GPU/Hızlandırıcı Desteği	Dolaylı (C/C++ GPU)	Evet (CUDA, ROCm -	Evet (GPU, TPU).	Hayır.	Hayır (ama süreçler kendi GPU	Hayır (ama süreçler kendi GPU

	kütüphane lerini çağırabilir).	gelişmekte e).			kullanımlarını yönetebilir).	kullanımları nı yönetebilir).
Otomatik Farklılaştır ma	Hayır.	Hayır.	Evet (temel özellik).	Hayır.	Hayır.	Hayır.
Uygulama Kolaylığı	Orta (ayrı “.pyx” dosyaları, “setup.py” gerekebili r).	Kolay (genellikl e sadece dekoratör).	Orta (yeni API, fonksiyone l paradigma) .	Kolay (üst düzey arayüz).	Kolay (üst düzey arayüz).	Orta (daha fazla kontrol, daha fazla karmaşıklık) .
Performans Kazanç Pot.	Yüksek	Yüksek	Çok Yüksek (özellikle hızlandırıcı larda)	Düşük (CPU- yoğun için), Yüksek (G/Ç- yoğun için)	Yüksek (CPU- yoğun için)	Yüksek (CPU-yoğun için)
Bellek Yönetimi	C seviyesind e kontrol, paylaşımlı bellek (dikkatli kullanım gerektirir) .	Genellikl e NumPy benzeri, paylaşımlı bellek (iş parçacıkla rı için).	XLA tarafından yönetilir, büyük tensörlere dikkat.	Paylaşımlı bellek (iş parçacıkları).	Ayrı bellek alanları (veri serileştirme/ deserileştirme gerekir).	Ayrı bellek alanları (veri serileştirme/ deserileştirm e ve IPC mekanizmal arı gerekir).
İletişim Yükü	Düşük (iş parçacıkla rı arasında).	Düşük (iş parçacıkla rı arasında).	Derlenmiş kerneller için	Düşük.	Orta (süreçler arası veri kopyalama).	Orta-Yüksek (seçilen IPC mekanizmas ına bağlı).

			optimize edilmiş.			
Entegrasyon	C/C++, NumPy.	NumPy.	NumPy API, TensorFlow (XLA).	Python standart kütüphanesi.	Python standart kütüphanesi.	Python standart kütüphânesi.

Tablo 23: Karşılaştırma Tablosu

Özellik	CUDA (Compute Unified Device Architecture)	ROCm (Radeon Open Compute platform)
Geliştirici	NVIDIA	AMD (Advanced Micro Devices)
Temel Amaç	NVIDIA GPU'ları üzerinde genel amaçlı paralel hesaplama için yazılım platformu ve programlama modeli.	AMD GPU'ları ve diğer hızlandırıcılar (potansiyel olarak) üzerinde yüksek performanslı hesaplama için açık kaynaklı yazılım platformu.
Donanım Desteği	Yalnızca NVIDIA GPU'ları.	Başlıca AMD Radeon Instinct ve bazı Radeon Pro/Gaming GPU'ları. Diğer mimarilere (örn: CPU, FPGA) genişletilebilirlik hedeflenir.
Programlama Dili	CUDA C/C++ (C/C++'ın uzantıları), Fortran desteği (PGI, vb.), Python arayüzleri (PyCUDA, Numba, CuPy).	HIP (Heterogeneous-compute Interface for Portability) - CUDA'ya çok benzer C++ API'si, OpenCL, Python arayüzleri (HIP Python, PyTorch, TensorFlow-ROCm).
Ekosistem ve Olgunluk	Çok olgun, geniş ve yerleşik bir ekosistem. Kapsamlı kütüphaneler (cuDNN, cuBLAS, cuFFT, Thrust, vb.), araçlar (Nsight) ve topluluk desteği.	Gelişmekte olan, ancak hızla büyüyen bir ekosistem. Önemli kütüphaneler (rocBLAS, rocFFT, MIOpen, rocThrust, vb.) mevcut, araç seti (ROCm Developer Tools) gelişiyor.
Açık Kaynak Durumu	Çoğunlukla kapalı kaynak (derleyici, sürücüler). Bazı kütüphaneler açık kaynak olabilir.	Büyük ölçüde açık kaynak (kernel sürücüsü, derleyici, kütüphaneler). Bu, daha fazla esneklik ve topluluk katkısı sağlar.

Taşınilabilirlik	Kodu doğrudan diğer GPU üreticilerine taşımak zordur (NVIDIA'ya özgü).	HIP sayesinde CUDA kodunun ROCm'a (ve tersi) taşınması nispeten kolaydır. Hedef, heterojen bilgi işlem için daha geniş taşınilabilirliktir.
Kütüphane Desteği	Çok geniş ve optimize edilmiş kütüphane seti. Derin öğrenme, lineer cebir, sinyal işleme vb. için olgun çözümler.	Temel matematik ve derin öğrenme kütüphaneleri mevcut ve gelişiyor. CUDA kütüphanelerinin kapsamına ve optimizasyon seviyesine ulaşmaya çalışıyor.
Derin Öğrenme Çerçeveleri Desteği	Tüm büyük çerçeveler (TensorFlow, PyTorch, MXNet, vb.) tarafından tam ve öncelikli destek.	TensorFlow ve PyTorch için resmi destek mevcut ve gelişiyor. Diğer çerçeveler için destek değişebilir veya topluluk tarafından sağlanabilir.
Kurulum ve Yapılandırma	Genellikle daha basit ve doğrudan, özellikle popüler Linux dağıtımlarında ve Windows'ta.	Biraz daha karmaşık olabilir, özellikle belirli GPU/İşletim Sistemi kombinasyonları için. Ancak son sürümlerde iyileşmeler var.
Performans	Yüksek kaliteli NVIDIA GPU'larında genellikle çok yüksek performans. Donanım ve yazılım sıkı entegrasyonu.	Yüksek kaliteli AMD GPU'larında rekabetçi performans. Optimizasyonlar ve yazılım desteği arttıkça performans da artıyor.
Lisanslama	NVIDIA'ya ait lisanslama.	Açık kaynak lisansları (MIT, Apache 2.0 vb.).
Topluluk Desteği	Çok büyük ve aktif bir geliştirici topluluğu, kapsamlı dokümantasyon ve forumlar.	Büyüyen bir topluluk, ancak CUDA kadar yaygın değil. Dokümantasyon ve kaynaklar artıyor.
Odak Noktası	Özellikle yapay zekâ/derin öğrenme, bilimsel hesaplama ve görselleştirme alanlarında güçlü.	Yüksek performanslı hesaplama (HPC), sunucu pazarı ve açık kaynak çözümlerine odaklanma.

Tablo 24: Karşılaştırma Tablosu

Özellik / Sürüm	Python 3.11	Python 3.12	Python 3.13 (Beklenen/Planlanan)	Python 3.14 (Beta - Beklenen/Planlanan)
Genel Performans	Önemli hız artışları (CPython için %10-60 ortalama, "Faster CPython" projesi). Daha hızlı başlangıç süresi.	Performans iyileştirmeleri devam etti (özellikle PEP 669 ile düşük etkili izleme). Daha iyi JIT derleyici ipuçları.	"Faster CPython" projesinin devamı. İsteğe bağlı JIT derleyicisi (PEP 744) üzerinde çalışmalar. GIL'in kaldırılması (Per-Interpreter GIL - PEP 684) veya daha granüler hale getirilmesi yönünde önemli adımlar atılabilir.	3.13'teki performans iyileştirmelerinin üzerine eklemeler. JIT derleyicisinin daha da geliştirilmesi ve GIL ile ilgili çalışmaların olgunlaşması beklenebilir.
Kuantum Kütüphane Uyumluluğu	Tüm büyük kuantum kütüphaneleri (Qiskit, Cirq, PennyLane vb.) tarafından tam desteklenir ve yaygın kullanılır.	Genellikle 3.11 çıktıktan kısa bir süre sonra büyük kuantum kütüphaneleri tarafından desteklenmeye başlandı.	Kararlı sürüm çıktığında, kuantum kütüphanelerinin hızla desteklenmesi beklenir.	Beta aşamasında kütüphane uyumluluğu sınırlı olabilir, ancak kararlı sürümle birlikte destek artacaktır.
Eşzamanlılık ve Paralellik	"asyncio" iyileştirmeleri devam etti. "TaskGroup" (PEP 654) eklendi.	"asyncio" ve "multiprocessing" modüllerinde iyileştirmeler.	GIL ile ilgili olası değişiklikler, CPU-yoğun görevlerde gerçek paralellik potansiyelini artırabilir.	GIL reformları olgunlaşırsa, "threading" modülünün CPU-yoğun işlerdeki etkinliği artabilir.

Tip İpuçları (Type Hinting)	"Self" tipi (PEP 673), "TypeVarTuple" (PEP 646) gibi önemli geliştirmeler.	"TypeIs" (PEP 647), "ParamSpec" ve "Concatenate" için daha iyi "infer_variance" (PEP 696) gibi iyileştirmeler.	Tip sistemi üzerinde daha fazla iyileştirme ve yeni özellikler bekleniyor.	Tip sisteminin daha da rafine edilmesi ve karmaşık senaryolar için daha iyi destek sunması beklenebilir.
Dil Özellikleri ve Kütüphane	"tomllib" (TOML ayırıştırma) eklendi. Hata mesajları iyileştirildi.	F-string'lerde iyileştirmeler (PEP 701). Linux "perf" profilleyici desteği (PEP 669).	Yeni standart kütüphane modülleri veya mevcut modüllerde önemli güncellemeler olabilir.	Yeni dil özellikleri veya standart kütüphane eklemeleri devam edebilir.
Kuantum Hesaplamaların Etkisi	Hız artışı, büyük kuantum simülasyonları ve optimizasyon görevleri için faydalıdır. Gelişmiş tip ipuçları, büyük kuantum projelerinde kod kalitesini artırır.	Performans ve tip sistemi iyileştirmeleri, 3.11'in faydalarını pekiştirir.	Özellikle GIL ile ilgili potansiyel değişiklikler, klasik ön/son işleme ve hibrit algoritmalarda kuantum hesaplamalarını ciddi şekilde hızlandırabilir. İsteğe bağlı JIT, sayısal olarak yoğun kuantum simülasyonlarını daha da hızlandırabilir.	3.13'teki olumlu etkilerin devamı. Daha olgun bir JIT ve GIL çözümü, kuantum simülasyonlarının ve algoritmalarının klasik kısımlarında devrim yaratabilir.
Kararlılık ve Topluluk Desteği	Kararlı, geniş topluluk desteği ve	Kararlı, topluluk desteği hızla artıyor.	Beta/RC aşamalarından sonra kararlı olacak.	Henüz beta aşamasında, kararlı kullanım için

	kaynaklar. Endüstri standartı.		Topluluk adaptasyonu zamanla artacak.	önerilmez. Topluluk adaptasyonu kararlı sürümden sonra başlar.
--	--------------------------------------	--	--	---

Tablo 25: Karşılaştırma Tablosu

Genel Değerlendirme:

Kuantum hesaplamaları için Python sürümü seçerken, genellikle en son **kararlı** sürüm veya bir önceki kararlı sürüm tercih edilir. Bu, hem en güncel performans iyileştirmelerinden faydalanmayı hem de kuantum kütüphanelerinin tam desteğini almayı sağlar.

- **Python 3.11:** Şu anda kuantum hesaplamaları için çok sağlam ve performanslı bir seçenektir. “Faster CPython” projesinin getirdiği hızlanmalar, simülasyon sürelerini olumlu etkiler.
- **Python 3.12:** 3.11 üzerine ek iyileştirmeler sunar ve kuantum kütüphaneleri tarafından genellikle iyi desteklenir. Yeni projeler için iyi bir adaydır.
- **Python 3.13:** Özellikle GIL’in kaldırılması veya daha granüler hale getirilmesi ve isteğe bağlı bir JIT derleyicisinin eklenmesi gibi potansiyel büyük değişiklikler vaat ediyor. Bu değişiklikler gerçekleşirse, kuantum simülasyonlarının CPU-yoğun kısımlarında ve hibrit algoritmalarda devrim niteliğinde performans artışları sağlayabilir. Ancak bu özelliklerin tam olarak nasıl şekilleneceği ve kuantum ekosistemine adaptasyonu zaman alacaktır.
- **Python 3.14 (Beta):** 3.13’teki devrimsel değişikliklerin üzerine inşa edilmesi ve bu özelliklerin daha da olgunlaştırılması beklenir. Kararlı hale geldiğinde, kuantum hesaplamaları için potansiyel olarak en iyi platform olabilir, ancak beta aşamasında üretim ortamları için uygun değildir.

Kuantum hesaplama geliştiricileri için en önemli faktör, kullandıkları kuantum kütüphanelerinin (Qiskit, Cirq, PennyLane, TensorFlow Quantum vb.) resmi olarak desteklediği Python sürümlerini takip etmektir. Genellikle bu kütüphaneler, yeni Python sürümlerine hızla adapte olurlar.

Özellik	Python	Julia	C++	Rust	Q# (Q Sharp)
---------	--------	-------	-----	------	--------------

Temel Tanım	Yüksek seviyeli, yorumlanan, genel amaçlı, geniş ekosistemli dil.	Yüksek seviyeli, dinamik, yüksek performanslı, bilimsel hesaplamaya odaklı dil.	Düşük/orta seviyeli, derlenen, yüksek performanslı, sistem programlama dili.	Düşük seviyeli, derlenen, bellek güvenliğine ve eşzamanlılığa odaklı sistem dili.	Microsoft tarafından kuantum algoritmalarını ifade etmek ve çalıştırmak için geliştirilmiş, üst düzey, alan odaklı dil.
Performans (Klasik Kısım)	Genellikle daha yavaş (saf Python), ancak C/C++ eklentileri (NumPy, SciPy) ve JIT derleyiciler (Numba, Jax) ile hızlandırılabilir.	C'ye yakın performans ("iki dil problemi"ni çözme) hedefler. JIT derleme.	Çok yüksek performans, donanıma yakın kontrol.	C++'a yakın performans, bellek güvenliği garantileriyle.	Klasik kontrol akışı Python veya .NET dillerinde çalışır; Q#'ın kendisi kuantum operasyonların tanımlar.
Kuantum Ekosistemi ve Kütüphaneler	Çok zengin: Qiskit (IBM), Cirq (Google), PennyLane (Xanadu), TensorFlow Quantum, PyQuil (Rigetti) gibi birçok büyük kütüphane.	Gelişmekte olan ekosistem: Yao.jl, QuantumOptics.jl, QuantumClifford.jl gibi kütüphaneler. Topluluk büyüyor.	Bazı özel simülatörler veya kütüphanelerin çekirdekleri C++ ile yazılmıştır. Qulacs gibi kütüphaneler C++ API sunar.	Sınırlı ancak artan ilgi. Bazı simülasyon projeleri Rust kullanıyor. Henüz büyük, yerleşik bir kuantum	Microsoft Quantum Development Kit (QDK) ile sıkı entegrasyon. QDK, simülatörler, kütüphaneler ve Azure Quantum'a erişim sağlar.

				kütüphanesi yok.	
Kullanım Kolaylığı ve Öğrenme Eğrisi	Öğrenmesi ve kullanması kolay, geniş topluluk ve kaynaklar.	Python'a benzer sözdizimi, ancak bazı farklılıklar. Matematiksel notasyona yakınlık.	Daha dik öğrenme eğrisi, manuel bellek yönetimi (modern C++ ile daha az).	Daha dik öğrenme eğrisi (sahiplik ve ödünç alma konseptleri nedeniyle).	Kuantum konseptlerine aşina olanlar için nispeten kolay. C#, F# ve Python ile benzerlikler taşır.
Bellek Yönetimi	Otomatik (çöp toplama).	Otomatik (çöp toplama).	Manuel (RAII ve akıllı işaretçilerle modern C++'da yönetimi kolaylaştırır).	Derleme zamanı bellek güvenliği (sahiplik ve ödünç alma), çöp toplayıcı yok.	QDK tarafından yönetilir; klasik kontrol kodu için temel dilin bellek yönetimi geçerlidir.
Eşzamanlılık ve Paralellik	"multiprocessing", "threading", "asyncio". GIL, CPU-yoğun işlerde "threading"i sınırlar.	Dahili çoklu iş parçacığı ve dağıtılmış hesaplama desteği.	Kapsamlı "std::thread", "std::async", OpenMP, MPI gibi kütüphanelerle destek.	Güçlü ve güvenli eşzamanlılık özellikleri (korkusuz eşzamanlılık).	Klasik kontrol akışında temel dilin yeteneklerine bağlıdır.
Kuantum Donanım Entegrasyonu	Büyük kütüphaneler aracılığıyla çeşitli donanım ve bulut	Bazı kütüphaneler aracılığıyla sınırlı entegrasyon, ancak gelişiyor.	Genellikle donanım sağlayıcılarının SDK'larının alt seviye	Doğrudan entegrasyon nadir, ancak teorik olarak mümkün.	Azure Quantum aracılığıyla çeşitli donanım

	platformlarına (IBM Quantum, Google Quantum AI, AWS Braket, Azure Quantum) erişim.		bileşenleri için kullanılır.		ortaklarına (IonQ, Quantinuum, Rigetti vb.) erişim.
Odak Alanı (Kuantumda)	Algoritma geliştirme, prototipleme, eğitim, hibrit kuantum-klasik algoritmalar, makine öğrenmesi.	Yüksek performanslı simülasyonlar, bilimsel hesaplama odaklı kuantum araştırmaları.	Yüksek performanslı simülatörler geliştirmek, kütüphane çekirdekleri oluşturmak.	Güvenli ve performanslı sistem düzeyinde kuantum araçları veya simülatörler geliştirmek.	Kuantum algoritmalarını üst düzeyde ifade etmek, kuantum donanımlarında çalıştırmak, kaynak tahmini yapmak.
Avantajları	Geniş ekosistem, hızlı prototipleme, öğrenme kolaylığı, büyük topluluk.	Yüksek performans, Python benzeri sözdizimi, iyi matematiksel yetenekler.	Maksimum performans ve kontrol, yerleşik sistemlerde kullanım.	Bellek güvenliği, performans, modern dil özellikleri, korkusuz eşzamanlılık.	Kuantum algoritmaların a odaklı tasarım, güçlü tip sistemi, donanım soyutlaması.
Dezavantajları	Saf Python'da düşük performans, GIL kısıtlaması.	Python'a göre daha küçük ekosistem ve topluluk.	Karmaşıklık, dik öğrenme eğrisi, manuel bellek yönetimi riskleri.	Daha dik öğrenme eğrisi, Python'a göre daha küçük kuantum ekosistemi.	Microsoft ekosistemine ve QDK'ya bağımlılık. Sadece kuantum algoritmaların a odaklı.

Tablo 26: Karşılaştırma Tablosu

Kuantum hesaplama için farklı programlama dillerini karşılaştırmak, her birinin sunduğu avantajları, dezavantajları ve ekosistemlerini anlamak açısından önemlidir.

Özet ve Seçim Kriterleri (Summary and Selection Criteria):

- **Python:** En yaygın başlangıç noktasıdır. Hızlı prototipleme, geniş kütüphane desteği ve eğitim materyalleri sayesinde kuantum algoritmalarını öğrenmek ve denemek için mükemmeldir. Performans kritik kısımlar için Cython, Numba gibi araçlarla veya C++ ile yazılmış kütüphanelerle desteklenir.
 - *Ideal for: Beginners, algorithm prototypers, researchers needing broad library support.*
- **Julia:** Python'un kullanım kolaylığı ile C'nin performansını birleştirmeyi hedefler. Özellikle yüksek performanslı bilimsel hesaplama ve simülasyon gerektiren kuantum araştırmaları için güçlü bir adaydır. Ekosistemi hızla büyümektedir.
 - *Ideal for: Researchers needing high performance for simulations without leaving a high-level language, scientific computing focus.*
- **C++:** Ham performansın ve donanım üzerinde tam kontrolün gerekli olduğu durumlar için tercih edilir. Genellikle yüksek performanslı kuantum simülatörlerinin veya kuantum kütüphanelerinin temelini oluşturmak için kullanılır.
 - *Ideal for: Developers of core simulation engines, performance-critical library components, low-level hardware interaction.*
- **Rust:** Bellek güvenliği ve korkusuz eşzamanlılık vaadiyle C++'a modern bir alternatif sunar. Kuantum alanındaki ekosistemi henüz başlangıç aşamasındadır ancak güvenli ve performanslı sistem düzeyinde araçlar geliştirmek için potansiyeli yüksektir.
 - *Ideal for: Developers prioritizing safety alongside performance for system-level tools or new simulation frameworks.*
- **Q#:** Özellikle Microsoft'un kuantum ekosistemiyle çalışmak ve kuantum algoritmalarını soyut bir şekilde ifade edip Azure Quantum gibi platformlarda çalıştırmak için tasarlanmıştır. Kuantum algoritmalarına odaklıdır ve klasik hesaplama kısımları için genellikle Python veya C# kullanılır.
 - *Ideal for: Users within the Microsoft quantum ecosystem, those focused on writing and running algorithms on target quantum hardware via Azure, resource estimation.*

Seçim, projenin gereksinimlerine, geliştiricinin aşinalığına, performans ihtiyaçlarına ve ekosistem desteğine bağlı olacaktır. Çoğu zaman, özellikle hibrit algoritmalarda bu dillerin bir kombinasyonu kullanılır (örneğin, Python ile üst düzey kontrol, C++ veya Julia ile performans kritik simülasyon çekirdekleri).

Örnek bir çıktı ve değerlendirme:

ThreadPoolExecutor Süresi: 9.1551 (Python 3.11), 8.8717–9.6249 (Python 3.13) saniye

Saf Python Süresi: 7.6752 saniye

Numba Süresi: 0.8431 saniye

Cython Süresi: **0.0280** saniye

Performans Sürelerinin Değerlendirilmesi

Süreler, belirli türdeki görevler için genellikle beklenen bir örüntüyü yansıtır, ancak ThreadPoolExecutor süresi biraz dikkat çekici.

- **Saf Python Süresi: 7.6752 saniye**

- Bu, yorumlanan bir dil olan Python'un doğal performansıdır. Özellikle CPU-yoğun (hesaplama ağırlıklı) görevlerde, derlenmiş dillere veya JIT (Just-In-Time) derleyicilere göre daha yavaştır. Bu temel çizginizdir.

- **ThreadPoolExecutor Süresi: 9.1551 (Python 3.11), 8.8717–9.6249 (Python 3.13) saniye**

- Bu sonucun saf Python'dan **daha yavaş** olması, üzerinde çalıştığınız görevin büyük olasılıkla **CPU-yoğun** bir görev olduğunu ve Global Interpreter Lock (GIL) nedeniyle gerçek iş parçacığı paralellikinden faydalanamadığını gösterir.
- ThreadPoolExecutor (ve threading modülü genel olarak), I/O-yoğun görevler (örneğin, ağ istekleri, dosya okuma/yazma gibi bekleme sürelerinin olduğu görevler) için daha uygundur. Çünkü bir iş parçacığı beklerken diğeri çalışabilir.
- CPU-yoğun görevlerde ise GIL, aynı anda sadece bir Python iş parçacığının Python bytecode'unu çalıştırmasına izin verir. Bu durumda, iş parçacıklarını yönetmenin getirdiği ek yük (thread oluşturma, context switching, senkronizasyon) saf Python'dan daha yavaş sonuçlara yol açabilir.

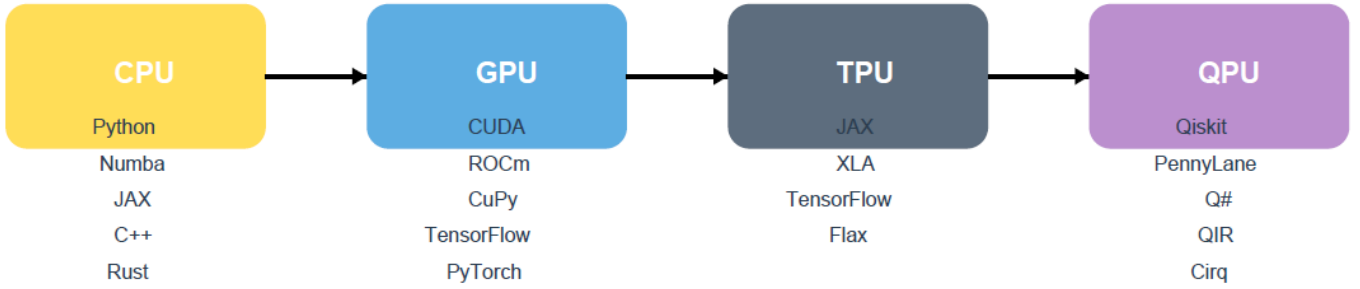
- Eğer göreviniz CPU-yoğun ise ve paralelleştirmek istiyorsanız, multiprocessing.Pool veya ProcessPoolExecutor kullanmak (ayrı işlemler, dolayısıyla ayrı GIL'ler) daha iyi sonuçlar verir.
- **Numba Süresi: 0.8431 saniye**
 - Bu, Numba'nın gücünü gösteriyor. Numba, Python fonksiyonlarını (özellikle NumPy kullanan ve sayısal hesaplama yapanları) çalışma zamanında makine koduna JIT derleyerek önemli ölçüde hızlandırır. Python'dan ~9 kat daha hızlı olması, görevinizin Numba'nın optimize edebileceği türden bir sayısal hesaplama içerdiğini düşündürür. İlk çalıştırmada bir miktar derleme süresi olabilir, sonraki çalıştırmalar daha da hızlı olabilir.
- **Cython Süresi: 0.0280 saniye**
 - Bu, **beklenen en iyi sonuçtur**. Cython, Python kodunu C/C++ koduna çevirir ve sonra bu kodu derler. Statik tiplendirme ve C seviyesinde optimizasyonlar yaparak Python'un yavaşlığını aşar. Saf Python'dan ~270 kat, Numba'dan ~30 kat daha hızlı olması, Cython'un bu tür bir görev için ne kadar etkili olabileceğini gösterir. Bu, muhtemelen C/C++ ile yazılmış bir kodun performansına çok yakındır.

Sonuç: Cython ve Numba'nın saf Python'a göre çok daha hızlı olması normal ve beklenendir. ThreadPoolExecutor'ın saf Python'dan yavaş olması, görevinizin CPU-yoğun olduğunu ve GIL kısıtlamasından etkilendiğini gösterir. Bu durum da CPU-yoğun görevler için normaldir.

IV. Hesaplama Paradigmalarının Tekâmülü: CPU -> GPU -> TPU -> QPU Yolculuğunda Programlama Dillerinin ve Modüllerin Geleceği

Aşama	Donanım	Programlama Dili / Araç
1	CPU	Python, C++, Rust, Numba, Jax
2	GPU	CUDA, ROCm, PyTorch, TensorFlow, CuPy
3	TPU	JAX, XLA, TensorFlow
4	QPU	Qiskit, PennyLane, Q#, QIR

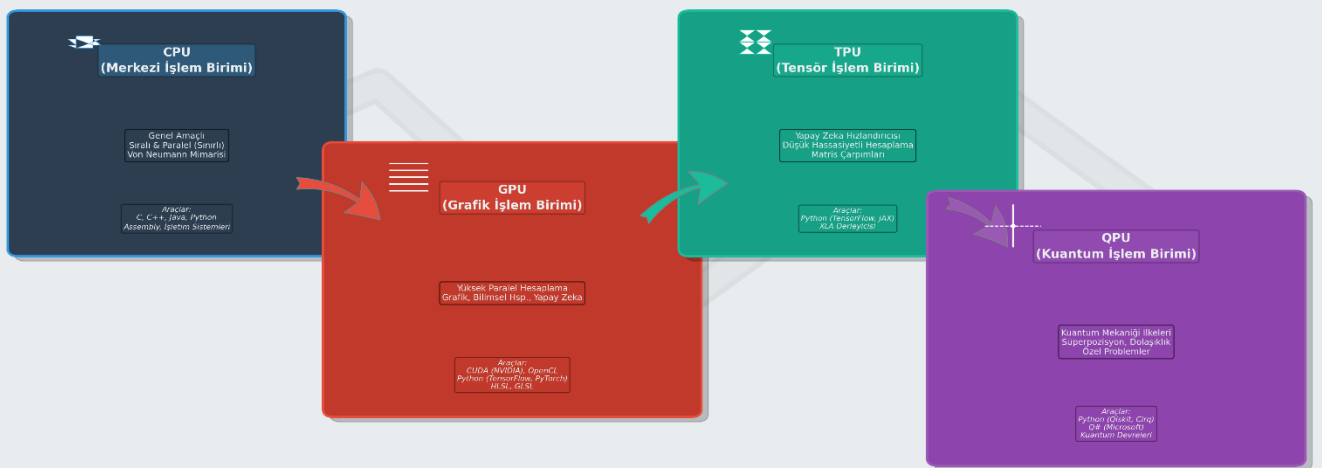
Tablo 27: CPU → QPU Tekâmülü



Şekil 116: Kuantum Hesaplamaya Giden Yol: Donanım & Yazılım Tekâmülü

- **CPU (Merkezi İşlem Birimi):** Klasik hesaplamanın genel amaçlı çalışan atıdır; karmaşık seri görevleri yerine getirir.
- **GPU (Grafik İşlem Birimi):** Grafik işlemeden evrilerek, büyük veri kümelerinin paralel işlenmesi için baskın mimari haline gelmiştir; yapay zekâ ve bilimsel hesaplama için çok önemlidir.
- **TPU (Tensor İşlem Birimi):** Google tarafından, sinir ağı makine öğrenimi iş yüklerini hızlandırmak için özel olarak tasarlanmış, özelleşmiş bir uygulamaya özel entegre devredir (ASIC).
- **QPU (Kuantum İşlem Birimi):** Klasik sistemler için çözülmesi zor olan belirli problem sınıflarını çözmek için süperpozisyon ve dolaşıklık gibi kuantum mekaniksel olgulardan yararlanan yeni ortaya çıkan paradigma.

Hesaplama Paradigmalarının Tekâmülü ve Programlama Araçları



CPU → GPU → TPU → QPU

Şekil 117: Hesaplama Paradigmalarının Tekâmülü ve Programlama Araçları

Hesaplama Çağları ve Programlama Araçları:

- **CPU Çağı:** o **Diller:** C, C++, Fortran, Java, Python. o **Paradigma:** Seri yürütme, karmaşık kontrol akışları ve genel amaçlı mantığa odaklanır. o **Ana Modüller/Kütüphaneler:** Standart kütüphaneler, NumPy (sayısal işlemler için), SciPy.
- **GPU Çağı:** o **Diller:** CUDA (Nvidia), HIP/ROCm (AMD), OpenCL (çoklu satıcı), SYCL. o **Soyutlama Katmanları:** Numba, PyTorch, TensorFlow, JAX (kodu GPU'larda çalıştırmak için derleyebilir). o **Paradigma:** Kitlesel paralel işleme. Programcının odağı, binlerce iş parçasığı üzerinde paralel olarak yürütülen çekirdekleri (kernel) tanımlamaya kayar.
- **TPU Çağı:** o **Diller/Çerçeveler:** Birincil arayüz, **JAX, PyTorch ve TensorFlow** gibi yüksek seviyeli çerçevelerdir. Bu çerçeveler kullanılarak Python'da yazılan kod, TPU donanımı üzerinde verimli çalışacak şekilde derlenir. o **Paradigma:** Doğrusal cebir ve derin öğrenme için alana özgü hesaplama. Soyutlama çok yüksektir; çerçeve ve derleyici, TPU'nun özelleşmiş sistolik dizi mimarisine düşük seviyeli eşleme işlemini yönetir.
- **QPU Çağı:** o **Diller:** Standart, hibrit kuantum-klasik dillerdir. * **Qiskit (Python SDK):** En yaygın kabul görmüş çerçevedir; kuantum devrelerinin Python içinde tanımlanmasına ve manipüle edilmesine olanak tanır. * **Cirq (Python):** Google'ın, NISQ işlemciler üzerinde kuantum devreleri tasarlamak, simüle etmek ve çalıştırmak için olan çerçevesidir. * **Q# (Microsoft):** Kuantum algoritmaları için özel olarak tasarlanmış, bağımsız bir dildir; genellikle klasik bir ana bilgisayar ile (örneğin, C# veya Python'da) kullanılır. o **Paradigma:** Hibrit kuantum-klasik hesaplama. Programlar kuantum devrelerini (geçitler, kübitler kullanarak) tanımlar ancak kontrol akışı, ön ve son işleme için ağırlıklı olarak klasik mantığa güvenir. Odak, kuantum avantajından yararlanan algoritmalar tasarlamaktır.

Târihsel Yolculuğun Özeti:

CPU'dan QPU'ya olan yörünge, donanımda **artan uzmanlaşma** ve dolayısıyla yazılımda **artan soyutlama** yönünde net bir eğilim göstermektedir. Programcılar, doğrudan donanım yönetiminden uzaklaşmaktadır. Gelecek, geliştiricilerin niyetlerini ifade etmelerine (örneğin, bir model eğitmek, bir kuantum kimyası simülasyonu çalıştırmak) izin verirken, sofistike derleyicilerin ve çalışma zamanı sistemlerinin bu niyeti, ister GPU, TPU isterse QPU olsun, benzersiz ve özelleşmiş temel donanıma eşleme gibi karmaşık görevi halletmesini sağlayan, alana özgü dillerde ve yüksek seviyeli çerçevelerde yatmaktadır.

Yirminci yüzyılın ortalarından itibaren dijital hesaplama, bilimsel keşiflerden günlük yaşama kadar toplumun her veçhesini dönüştürmüştür. Bu dönüşümün merkezinde, giderek daha karmaşık ve güçlü hale gelen işlem birimleri yer almaktadır. Merkezi İşlem Birimleri (CPU'lar) ile başlayan bu yolculuk, Grafik İşlem Birimleri (GPU'lar) ve Tensör İşlem Birimleri (TPU'lar) gibi özel amaçlı hızlandırıcılarla devam etmiş ve şimdi de Kuantum İşlem Birimleri (QPU'lar) ile yeni bir devrimin eşiğindedir. Donanımdaki bu gelişim, kaçınılmaz olarak yazılım geliştirme paradigmalarını, programlama dillerini ve kullanılan araçları da derinden etkilemektedir. Bu makale, CPU'dan QPU'ya uzanan bu teknolojik ilerlemenin programlama dünyasındaki yansımalarını ve gelecekteki potansiyel yönelimleri akademik bir perspektifle inceleyecektir.

CPU Çağı: Genel Amaçlı Hesaplamanın Temelleri CPU'lar, on yıllardır hesaplamanın temel taşı olmuş, çok çeşitli görevleri yerine getirebilen genel amaçlı işlemcilerdir. Bu dönemde geliştirilen programlama dilleri (C, C++, Java, Python gibi) ve işletim sistemleri, öncelikle sıralı işlem mantığına ve daha sonra çoklu görev (multitasking) ile sınırlı paralellığe (multithreading) odaklanmıştır. Python gibi dillerde Global Interpreter Lock (GIL) gibi mekanizmalar, CPU-yoğun görevlerde gerçek iş parçacığı paralellliğini kısıtlamış, bu da geliştiricileri alternatif çözümler aramaya itmiştir. Bu dönemin temel zorlukları arasında algoritma verimliliği, bellek yönetimi ve tek bir işlemcinin sınırlarını zorlayan görevler için ölçeklenebilirlik yer almıştır. Derleyiciler ve yorumlayıcılar, insan tarafından okunabilir kodu makine diline çevirmede kritik rol oynamıştır.

GPU Devrimi: Paralel Hesaplamanın Yükselişi Başlangıçta grafik işleme için tasarlanan GPU'lar, binlerce küçük çekirdeğe sahip olmaları sayesinde devasa paralel hesaplama yetenekleri sunmuştur. Bu potansiyel, bilimsel hesaplama, finansal modelleme ve özellikle yapay zeka (AI) ve makine öğrenimi (ML) alanlarında keşfedilmiştir. CUDA (Compute Unified Device Architecture) platformu ve açık standart olan OpenCL, GPU programlamada öncü rol oynamıştır. Bu diller/API'ler, geliştiricilerin C/C++ benzeri bir sözdizimiyle GPU çekirdeklerini (kernel'ları) doğrudan programlamasına olanak tanımıştır. Ancak, bu düşük seviyeli programlama karmaşık ve hataya açıktır. Bu zorluğu aşmak için TensorFlow, PyTorch, JAX (Python tabanlı) gibi yüksek seviyeli kütüphaneler ve çatılar (frameworks) geliştirilmiştir. Bu araçlar, GPU programlamanın karmaşıklığını soyutlayarak, veri bilimcilerin ve ML mühendislerinin Python gibi daha erişilebilir dillerle GPU'ların gücünden faydalanmasını sağlamıştır. Arka planda, bu kütüphaneler optimize edilmiş GPU kernel'larını (cuDNN, cuBLAS gibi) kullanarak performansı maksimize eder. GPU programlamanın temel zorlukları arasında veri transfer darboğazları (CPU-GPU arası), bellek hiyerarşisi

yönetimi ve kernel optimizasyonu yer almaktadır. Gelecekte, GPU mimarilerinin daha da özelleşmesi ve programlama modellerinin daha yüksek soyutlama seviyelerine ulaşması beklenmektedir.

TPU'lar ve AI Hızlandırıcıları: Özelleşmenin Derinleşmesi Makine öğrenimi, özellikle derin öğrenme modellerinin büyümesiyle birlikte, TPU'lar gibi özel donanımlar (ASIC'ler) geliştirilmiştir. Google tarafından tasarlanan TPU'lar, tensör operasyonlarını (özellikle matris çarpımlarını) düşük hassasiyetli sayılarla (örneğin, bfloat16) son derece verimli bir şekilde gerçekleştirmek üzere optimize edilmiştir. Bu, hem performansı artırır hem de enerji tüketimini azaltır. TPU programlama genellikle TensorFlow ve JAX gibi Google destekli çatılar aracılığıyla yapılır. XLA (Accelerated Linear Algebra) derleyicisi, bu çatılardan gelen hesaplama grafiklerini optimize ederek TPU'larda verimli bir şekilde çalıştırır. TPU'ların ve benzeri AI hızlandırıcılarının (örneğin, Intel Nervana, Graphcore IPU) yükselişi, donanım-yazılım ortak tasarımının (co-design) önemini vurgulamaktadır. Gelecekte, bu tür hızlandırıcılar için daha standartlaşmış programlama arayüzleri ve farklı donanımlar arasında taşınabilirliği artıran derleyici teknolojileri kritik olacaktır. Python, bu ekosistemde de kullanıcı dostu bir arayüz sağlamaya devam edecektir, ancak altta yatan mekanizmalar giderek daha karmaşık hale gelecektir.

QPU Çağı: Kuantum Hesaplamanın Ufukları Kuantum hesaplama, klasik hesaplamanın temel ilkelerinden (bitler) farklı olarak kubitleri (qubit) kullanır. Kubitler, süperpozisyon (aynı anda 0 ve 1 olma) ve dolaşıklık (entanglement) gibi kuantum mekaniksel fenomenlerden yararlanarak belirli türdeki problemleri (örneğin, şifre kırma, malzeme bilimi, ilaç keşfi, optimizasyon) klasik bilgisayarların çözemeyeceği hızlarda çözme potansiyeli sunar. Kuantum programlama henüz emekleme aşamasındadır. IBM'in Qiskit (Python tabanlı), Google'ın Cirq (Python tabanlı) ve Microsoft'un Q# (bağımsız bir dil) gibi SDK'ları ve dilleri, geliştiricilerin kuantum algoritmaları tasarlamasına ve bunları simülatörlerde veya mevcut (gürültülü, orta ölçekli) kuantum donanımlarında çalıştırmasına olanak tanır. Kuantum programlama, klasik programlamadan farklı bir düşünme biçimi gerektirir; olasılıksal sonuçlar, kubitlerin hassasiyeti (decoherence) ve hata düzeltme mekanizmaları gibi yeni zorluklar sunar. Gelecekte, kuantum dillerinin daha yüksek seviyeli soyutlamalar sunması, klasik ve kuantum hesaplamayı hibrit bir şekilde entegre eden araçların geliştirilmesi ve kuantum hata düzeltme kodlarının daha etkin bir şekilde uygulanması beklenmektedir. Ayrıca, farklı kuantum donanım platformları arasında birlikte çalışabilirliği sağlayacak standartların oluşması da önemlidir. Python'ın bu alanda da birleştirici bir rol oynaması muhtemeldir.

Disiplinlerarası Temalar ve Gelecek Yönelimleri CPU'dan QPU'ya uzanan bu yolculukta birkaç önemli tema öne çıkmaktadır:

1. **Soyutlama Seviyeleri:** Donanım karmaşıklığı arttıkça, programcıları bu karmaşıklıktan koruyan ve üretkenliği artıran daha yüksek seviyeli soyutlama katmanlarına (örneğin, Python kütüphaneleri, DSL'ler) ihtiyaç duyulmaktadır.
2. **Derleyici Teknolojileri:** Heterojen sistemlerde (CPU+GPU+TPU) kodu verimli bir şekilde çalıştırmak ve farklı donanım hedeflerine optimize etmek için gelişmiş derleyici teknolojileri (örneğin, MLIR, XLA) hayati öneme sahiptir.
3. **Donanım-Yazılım Ortak Tasarımı:** Optimal performans için donanım mimarisi ve yazılım araçlarının birlikte tasarlanması giderek daha yaygınlaşmaktadır.
4. **Taşınabilirlik ve Birlikte Çalışabilirlik:** Geliştiricilerin kodlarını farklı donanım platformlarında minimum değişiklikle çalıştırabilmesi önemlidir. OpenCL, SYCL gibi standartlar bu yönde atılmış adımlardır.
5. **Eğitim ve Yetenek Gelişimi:** Her yeni donanım paradigması, geliştiricilerin yeni beceriler ve düşünme biçimleri edinmesini gerektirir. Özellikle kuantum programlama, bu konuda önemli bir zorluk teşkil etmektedir.
6. **Açık Kaynak Ekosistemleri:** TensorFlow, PyTorch, Qiskit gibi açık kaynaklı projeler, inovasyonu hızlandırmakta ve geniş bir kullanıcı topluluğunun oluşmasını sağlamaktadır.

Hesaplama donanımlarındaki tekâmül, programlama dillerinin ve araçlarının sürekli olarak yeniden şekillenmesini zorunlu kılmaktadır. CPU'ların genel amaçlı hakimiyetinden, GPU ve TPU'ların özel amaçlı hızlandırmasına ve şimdi de QPU'ların devrimsel potansiyeline doğru ilerlerken, yazılım dünyası da bu değişime ayak uydurmaktadır. Python gibi yüksek seviyeli diller, farklı donanım platformları için birleştirici bir arayüz sunarak karmaşıklığı yönetmede önemli bir rol oynamaktadır. Gelecekte, daha sofistike derleyiciler, daha yüksek soyutlama seviyeleri ve donanım-yazılım ortak tasarımına dayalı yaklaşımlar, bu güçlü ve çeşitli hesaplama kaynaklarından tam olarak yararlanmamızı sağlayacaktır. Bu yolculuk hem zorluklar hem de heyecan verici fırsatlarla dolu olup, bilişim teknolojilerinin sınırlarını genişletmeye devam edecektir. Daha önceki makalelerde [1–118, 119, 120, 121–355] bunlara örnekler vermiştim.

Not: The references provided here correspond to the first part of a prior study and are therefore not numbered or ordered based on their appearance in this paper. Additional citations introduced after the original submission are also omitted from the main reference list for consistency [322–480].

IV. References

1. Xu, S.-Y., Belopolski, I., Alidoust, N., et al. (2015). Discovery of a Weyl fermion semimetal and topological Fermi arcs. *Science*, 349(6248), 613–617. <https://doi.org/10.1126/science.aaa9297>
2. Hasan, M. Z., & Kane, C. L. (2010). Colloquium: Topological insulators. *Reviews of Modern Physics*, 82(3), 3045. <https://doi.org/10.1103/RevModPhys.82.3045>
3. Erhard, A., Wallman, J. J., Postler, L., et al. (2019). Characterizing large-scale quantum computers via cycle benchmarking. *Nature Communications*, 10(5347). <https://doi.org/10.1038/s41467-019-13068-7>
4. Lian, B., Sun, X.-Q., et al. (2018). Topological quantum computation based on chiral Majorana fermions. *Proceedings of the National Academy of Sciences*, 115(43), 10938–10942. <https://doi.org/10.1073/pnas.1810003115>
5. Motome, Y., & Nasu, J. (2019). Hunting Majorana fermions in Kitaev magnets. *Journal of the Physical Society of Japan*, 89(1), 012002. [arXiv:1909.02234v2] <https://doi.org/10.7566/JPSJ.89.012002>
6. Osterhage, W. W. (2020). Quantum computers. In *Mathematical Theory of Advanced Computing* (pp. 103–107). Springer. https://doi.org/10.1007/978-3-662-60359-8_7
7. Dirac, P. A. M. (1928). The quantum theory of the electron. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 117(778), 610–624. <https://doi.org/10.1098/rspa.1928.0023>
8. Ciudad, D. (2015). Massless yet real. *Nature Materials*, 14(863). <https://doi.org/10.1038/nmat4411>
9. Weyl, H. (1929). Elektron und Gravitation. I. *Zeitschrift für Physik*, 56(5–6), 330–352. <https://doi.org/10.1007/BF01339504>
10. Majorana, E. (1937). Teoria simmetrica dell'elettrone e del positrone. *Il Nuovo Cimento*, 14(4), 171–184. <https://doi.org/10.1007/BF02961314>
11. Herring, C. (1937). Accidental degeneracy in the energy bands of crystals. *Physical Review*, 52(4), 365–373. <https://doi.org/10.1103/PhysRev.52.365>
12. Xu, S.-Y., Alidoust, N., Chang, G., et al. (2017). Discovery of Lorentz-violating type II Weyl fermions in LaAlGe. *Science Advances*, 3(6), e1603266. <https://doi.org/10.1126/sciadv.1603266>
13. Huang, S.-M., Xu, S.-Y., Belopolski, I., et al. (2015). A Weyl fermion semimetal with surface Fermi arcs in the transition metal monpnictide TaAs class. *Nature Communications*, 6(7373). <https://doi.org/10.1038/ncomms8373>
14. Lu, L., Wang, Z., Ye, D., et al. (2015). Experimental observation of Weyl points. *Science*, 349(6248), 622–624. <https://doi.org/10.1126/science.aaa9273>
15. Lv, B. Q., Weng, H. M., et al. (2015). Experimental discovery of Weyl semimetal TaAs. *Physical Review X*, 5(3), 031013. <https://doi.org/10.1103/PhysRevX.5.031013>
16. Suzuki, Y. (2019). The Super-Kamiokande experiment. *The European Physical Journal C*, 79(4), 298. <https://doi.org/10.1140/epjc/s10052-019-6796-2>

17. Wang, J., Lian, B., & Zhang, S.-C. (2015). Quantum anomalous Hall effect in magnetic topological insulators. *Physica Scripta*, 2015(T164), 014003.
<https://doi.org/10.1088/0031-8949/2015/T164/014003>
18. Chang, C.-Z., et al. (2013). Experimental observation of the quantum anomalous Hall effect in a magnetic topological insulator. *Science*, 340(6129), 167–170.
<https://doi.org/10.1126/science.1234414>
19. Sinitsyn, N. A. (2007). Semiclassical theories of the anomalous Hall effect. *Journal of Physics: Condensed Matter*, 20(2), 023201. <https://doi.org/10.1088/0953-8984/20/02/023201>
20. Hall, E. H. (1879). On a new action of the magnet on electric currents. *American Journal of Mathematics*, 2(3), 287–292. <https://doi.org/10.2307/2369245>
21. Hall, E. H. (1881). On the “rotational coefficient” in nickel and cobalt. *Philosophical Magazine and Journal of Science*, Series 5, 12(74), 157–172. <https://doi.org/10.1080/14786448108627086>
22. Nagaosa, N., Sinova, J., et al. (2010). Anomalous Hall effect. *Reviews of Modern Physics*, 82(2), 1539. [arXiv:0904.4154v1] <https://doi.org/10.1103/RevModPhys.82.1539>
23. Moreau, P.-A., Toninelli, E., et al. (2019). Imaging Bell-type nonlocal behavior. *Science Advances*, 5(7), eaaw2563. <https://doi.org/10.1126/sciadv.aaw2563>
24. Cross, A. W., Bishop, L. S., et al. (2017). Open quantum assembly language. [arXiv:1707.03429]. <https://doi.org/10.48550/arXiv.1707.03429>
25. Bergholm, V., Izaac, J., et al. (2018). PennyLane: Automatic differentiation of hybrid quantum-classical computations. [arXiv:1811.04968] <https://doi.org/10.48550/arXiv.1811.04968>
26. Kitaev, A. Yu. (2001). Unpaired Majorana fermions in quantum wires. *Physics-Uspekhi*, 44(10S), 131–136. <https://doi.org/10.1070/1063-7869/44/10S/S29>
27. Gül, Ö., Zhang, H., Bommer, J. D. S., et al. (2018). Ballistic Majorana nanowire devices. *Nature Nanotechnology*, 13(3), 192–197. <https://doi.org/10.1038/s41565-017-0032-8>
28. Yesilyurt, C., Tan, S., Liang, G., et al. (2016). Klein tunneling in Weyl semimetals under the influence of magnetic field. *Scientific Reports*, 6, 38862. <https://doi.org/10.1038/srep38862>
29. Takane, D., Wang, Z., Souma, S., et al. (2019). Observation of chiral fermions with a large topological charge and associated Fermi-arc surface states in CoSi. *Physical Review Letters*, 122(7), 076402. <https://doi.org/10.1103/PhysRevLett.122.076402>
30. Tang, P., Zhou, Q., & Zhang, S.-C. (2017). Multiple types of topological fermions in transition metal silicides. *Physical Review Letters*, 119(20), 206402. <https://doi.org/10.1103/PhysRevLett.119.206402>
31. Stenger, J. P. T., & Mong, R. S. K. (2020). One-dimensional error-correcting code for Majorana qubits. *Physical Review A*, 101(4), 042338. <https://doi.org/10.1103/PhysRevA.101.042338>
32. Devitt, S. J., et al. (2013). Quantum error correction for beginners. *Reports on Progress in Physics*, 76(7), 076001. [arXiv:0905.2794v4] <https://doi.org/10.1088/0034-4885/76/7/076001>
33. Gottesman, D. (2009). An introduction to quantum error correction and fault-tolerant quantum computation. [arXiv:0904.2557] <https://doi.org/10.48550/arXiv.0904.2557>
34. Fukui, K., Tomita, A., & Okamoto, A. (2018). Tracking quantum error correction. *Physical Review A*, 98(2), 022326. <https://doi.org/10.1103/PhysRevA.98.022326>

35. Tzitrin, I., Bourassa, J. E., Menicucci, N. C., & Sabapathy, K. K. (2019). Towards practical qubit computation using approximate error-correcting grid states. [arXiv:1910.03673]
<https://doi.org/10.1103/PhysRevA.101.032315>
36. Song, C., Cui, J., Wang, H., et al. (2019). Quantum computation with universal error mitigation on a superconducting quantum processor. *Science Advances*, 5(9), eaaw5686.
<https://doi.org/10.1126/sciadv.aaw5686>
37. Preskill, J. (2018). Quantum computing in the NISQ era and beyond. [arXiv:1801.00862]
<https://doi.org/10.22331/q-2018-08-06-79>
38. Ferracin, S., et al. (2019). Accrediting outputs of noisy intermediate-scale quantum computing devices. *New Journal of Physics*, 21(11), 113038. <https://doi.org/10.1088/1367-2630/ab4fd6>
39. Emerson, J., Silva, M., Moussa, O., et al. (2008). *Science*, 317(5846), 1893–1896.
<https://doi.org/10.1126/science.1145699>
40. Otrokov, M. M., Klimovskikh, I. I., et al. (2019). Prediction and observation of an antiferromagnetic topological insulator. *Nature*, 576, 416–422. <https://doi.org/10.1038/s41586-019-1840-9>
41. Kaushal, V., Lekitsch, B., Stahl, A., et al. (2019). Shuttling-based trapped-ion quantum information processing. [arXiv:1912.04712] <https://doi.org/10.48550/arXiv.1912.04712>
42. Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (pp. 124–134). IEEE. [arXiv:quant-ph/9508027] <https://doi.org/10.1109/sfcs.1994.365700>
43. Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* (pp. 212–219). ACM. <https://doi.org/10.1145/237814.237866>
44. Simon, D. R. (1994). On the power of quantum computation. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (pp. 116–123). IEEE. & Simon, D. R. (1997). *SIAM Journal on Computing*, 26(5), 1474–1484. <https://doi.org/10.1137/S0097539796298637>
45. Deutsch, D., & Jozsa, R. (1992). Rapid solution of problems by quantum computation. *Proceedings of the Royal Society A*, 439(1907), 553–558. <https://doi.org/10.1098/rspa.1992.0167>
46. Cerasoli, F. T., Sherbert, K., Sławińska, J., & Nardelli, M. B. (2020). Quantum computation of silicon electronic band structure. *Physical Chemistry Chemical Physics*, 22(39), 21816–21822. [arXiv:2006.03807] <https://doi.org/10.1039/D0CP04008H>
47. Hanoyamak, T., & Chehraz, A. (2019). Fundamental structure of Shor's quantum algorithm for factoring integers. *Turkish Journal of Mathematics and Computer Science*, 11(2), 78–83. <https://dergipark.org.tr/en/pub/tjmcs/issue/51518/570297>
48. Strubell, E. (2011). An introduction to quantum algorithms.
<http://mmrc.amss.cas.cn/tlb/201702/W020170224608150507023.pdf>
49. Rosenberg, D., Lita, A. E., Miller, A. J., & Nam, S. W. (2005). Noise-free high-efficiency photon-number-resolving detectors. *Physical Review A*, 71(6), 061803(R). [arXiv:quant-ph/0506175]
<https://doi.org/10.1103/PhysRevA.71.061803>
50. Kotsubo, V., Radebaugh, R., Hendershott, P., et al. (2017). Compact 2.2 K cooling system for superconducting nanowire single photon detectors. *IEEE Transactions on Applied Superconductivity*.
<https://doi.org/10.1109/TASC.2017.2657682>

51. Singh, A., Li, Q., Liu, S., et al. (2019). Quantum frequency conversion of a quantum dot single-photon source on a nanophotonic chip. *Optica*, 6, 563–569. <https://doi.org/10.1364/OPTICA.6.000563>
52. Nishida, K., Taguchi, K., & Matsumoto, Y. (1979). InGaAsP heterostructure avalanche photodiodes with high avalanche gain. *Applied Physics Letters*, 35(7), 251. <https://doi.org/10.1063/1.91089>
53. Kim, S., & Marino, A. M. (2019). Atomic resonant single-mode squeezed light from four-wave mixing through feedforward. *Optics Letters*, 44(18), 4630–4633. <https://doi.org/10.1364/OL.44.004630>
54. Vaidya, V. D., Morrison, B., Helt, L. G., et al. (2020). Broadband quadrature-squeezed vacuum and nonclassical photon number correlations from a nanophotonic device. *Science Advances*, 6(39), aba9186. <https://doi.org/10.1126/sciadv.aba9186>
55. Caves, C. M. (1980). Quantum-mechanical radiation-pressure fluctuations in an interferometer. *Physical Review Letters*, 45(2), 75–79. <https://doi.org/10.1103/PhysRevLett.45.75>
56. Shor, P. W. (1996). Fault-tolerant quantum computation. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science* (pp. 56–65). IEEE Computer Society. [arXiv:quant-ph/9605011] <https://doi.org/10.48550/arXiv.quant-ph/9605011>
57. Franz, M. (2013). Majorana's wires. *Nature Nanotechnology*, 8(3), 149–152. [arXiv:1302.3641v2] <https://doi.org/10.1038/nnano.2013.33>
58. Nayak, C., Simon, S. H., Stern, A., et al. (2008). Non-Abelian anyons and topological quantum computation. *Reviews of Modern Physics*, 80(3), 1083–1159. <https://doi.org/10.1103/RevModPhys.80.1083>
59. Stern, A. (2010). Non-Abelian states of matter. *Nature*, 464(7286), 187–193. <https://doi.org/10.1038/nature08915>
60. Nakamura, J., Liang, S., Gardner, G. C., et al. (2020). Direct observation of anyonic braiding statistics. *Nature Physics*, 16(9), 931–936. [arXiv:2006.14115v1] <https://doi.org/10.1038/s41567-020-1019-1>
61. Wilczek, F. (Ed.). (1990). *Fractional statistics and anyon superconductivity*. World Scientific. ISBN 978-9810200497. <https://doi.org/10.1142/0961>
62. Venema, L., Verberck, B., Georgescu, I., et al. (2016). The quasiparticle zoo. *Nature Physics*, 12(12), 1085–1089. <https://doi.org/10.1038/nphys3977>
63. Willett, R. L., Nayak, C., Shtengel, K., et al. (2013). Magnetic-field-tuned Aharonov-Bohm oscillations and evidence for non-Abelian anyons at $\nu = 5/2$. *Physical Review Letters*, 111(18), 186401. <https://doi.org/10.1103/PhysRevLett.111.186401>
64. Bartolomei, H., et al. (2020). Fractional statistics in anyon collisions. *Science*, 368(6487), 173–177. [arXiv:2006.13157v1] <https://doi.org/10.1126/science.aaz5601>
65. Deutsch, D. (1985). Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society A*, 400(1818), 97–117. <https://doi.org/10.1098/rspa.1985.0070>
66. Loceff, M. (2015). *A course in quantum computing for the community college* (Vol. 1, No. 18, p. 362). http://lapastillaroja.net/wp-content/uploads/2016/09/Intro_to_QC_Vol_1_Loceff.pdf
67. Arora, S., & Barak, B. (2009). *Computational complexity: A modern approach*. Cambridge University Press. <https://doi.org/10.1017/CBO9780511804090>

68. Koiran, P., Nesme, V., & Portier, N. (2005). A quantum lower bound for the query complexity of Simon's problem. In *Automata, Languages and Programming* (pp. 1287–1298). Springer. https://doi.org/10.1007/11523468_104
69. Lerner, I. V., Altshuler, B. L., & Gefen, Y. (Eds.). (2004). *Fundamental problems of mesoscopic physics: Interactions and decoherence* (NATO Science Series II: Mathematics, Physics and Chemistry; Vol. 154). Springer. <https://doi.org/10.1007/1-4020-2193-3>
70. Taniguchi, N. (1974). On the basic concept of nano-technology. In *Proceedings of the International Conference on Production Engineering* (Tokyo, Part II). Japan Society of Precision Engineering.
71. Mantina, M., Chamberlin, A. C., Valero, R., et al. (2009). Consistent van der Waals radii for the whole main group. *Journal of Physical Chemistry A*, 113(19), 5806–5812. <https://doi.org/10.1021/jp8111556>
72. Iijima, S. (1991). Helical microtubules of graphitic carbon. *Nature*, 354(6348), 56–58. <https://doi.org/10.1038/354056a0>
73. Lin, J., He, C., Zhang, L., & Zhang, S. (2009). Sensitive amperometric immunosensor for α -fetoprotein based on carbon nanotube/gold nanoparticle doped chitosan film. *Analytical Biochemistry*, 384(1), 130–135. <https://doi.org/10.1016/j.ab.2008.09.033>
74. Josephson, B. D. (1962). Possible new effects in superconductive tunnelling. *Physics Letters*, 1(7), 251–253. [https://doi.org/10.1016/0031-9163\(62\)91369-0](https://doi.org/10.1016/0031-9163(62)91369-0)
75. Mineev, Z. K., Leghtas, Z., Mundhada, S. O., et al. (2021). Energy-participation quantization of Josephson circuits. *npj Quantum Information*, 7(1), 131. <https://doi.org/10.1038/s41534-021-00461-8>
76. Shnirman, A., Schoen, G., & Hermon, Z. (1997). Quantum manipulations of small Josephson junctions. *Physical Review Letters*, 79(13), 2371–2374. <https://doi.org/10.1103/PhysRevLett.79.2371>
77. Pekker, D., Hou, C.-Y., Manucharyan, V. E., & Demler, E. (2013). Proposal for coherent coupling of Majorana zero modes and superconducting qubits using the 4π Josephson effect. *Physical Review Letters*, 111(10), 107007. <https://doi.org/10.1103/PhysRevLett.111.107007>
78. Boehm, H. P. (1997). The first observation of carbon nanotubes. *Carbon*, 35(4), 581–584. [https://doi.org/10.1016/S0008-6223\(97\)83730-X](https://doi.org/10.1016/S0008-6223(97)83730-X)
79. Iijima, S., & Ichihashi, T. (1993). Single-shell carbon nanotubes of 1-nm diameter. *Nature*, 363(6430), 603–605. <https://doi.org/10.1038/363603a0>
80. Bethune, D. S., Kiang, C. H., De Vries, M. S., et al. (1993). Cobalt-catalysed growth of carbon nanotubes with single-atomic-layer walls. *Nature*, 363(6430), 605–607. <https://doi.org/10.1038/363605a0>
81. Abdullayeva, S. H., Huseynov, A. B., Musayeva, N. N., et al. (2016). Synthesis of carbon nanotubes using Azerbaijan's oil. *Advances in Materials Physics and Chemistry*, 6(5), 105–112. <https://doi.org/10.4236/ampc.2016.65011>
82. Hutchison, J. L., Kiselev, N. A., Krinichnaya, E. P., et al. (2001). Double-walled carbon nanotubes fabricated by a hydrogen arc discharge method. *Carbon*, 39(5), 761–770. [https://doi.org/10.1016/S0008-6223\(00\)00187-1](https://doi.org/10.1016/S0008-6223(00)00187-1)
83. Shi, Z., Lian, Y., Liao, F. H., et al. (2000). Large scale synthesis of single-wall carbon nanotubes by arc-discharge method. *Journal of Physics and Chemistry of Solids*, 61(7), 1031–1036. [https://doi.org/10.1016/S0022-3697\(99\)00358-3](https://doi.org/10.1016/S0022-3697(99)00358-3)

84. Sugai, T., Yoshida, H., Shimada, T., et al. (2003). New synthesis of high-quality double-walled carbon nanotubes by high-temperature pulsed arc discharge. *Nano Letters*, 3(6), 769–773. <https://doi.org/10.1021/nl034183>
85. Vander Wal, R. L., Berger, G. M., & Ticich, T. M. (2003). Carbon nanotube synthesis in a flame using laser ablation for in situ catalyst generation. *Applied Physics A*, 77(7), 885–889. <https://doi.org/10.1007/s00339-003-2196-3>
86. Chen, C., Chen, W., & Zhang, Y. (2005). Synthesis of carbon nano-tubes by pulsed laser ablation at normal pressure in metal nano-sol. *Physica E: Low-Dimensional Systems and Nanostructures*, 28(1–2), 121–127. <https://doi.org/10.1016/j.physe.2005.02.006>
87. Koziol, K., Boskovic, B. O., & Yahya, N. (2010). Synthesis of carbon nanostructures by CVD method. In *Carbon and oxide nanostructures* (pp. 23–48). Springer. https://doi.org/10.1007/8611_2010_12
88. Meysami, S. S., Dillon, F., Koos, A. A., & Grobert, N. (2013). Aerosol-assisted chemical vapour deposition synthesis of multi-wall carbon nanotubes: I. Mapping the reactor. *Carbon*, 58, 151–158. <https://doi.org/10.1016/j.carbon.2013.02.044>
89. Meysami, S. S., Koos, A. A., Dillon, F., & Grobert, N. (2013). Aerosol-assisted chemical vapour deposition synthesis of multi-wall carbon nanotubes: II. An analytical study. *Carbon*, 58, 159–169. <https://doi.org/10.1016/j.carbon.2013.02.041>
90. Meysami, S. S., Koos, A. A., Dillon, F., Dutta, M., & Grobert, N. (2015). Aerosol-assisted chemical vapour deposition synthesis of multi-wall carbon nanotubes: III. Towards upscaling. *Carbon*, 88, 148–156. <https://doi.org/10.1016/j.carbon.2015.02.045>
91. Nasibulin, A. G., Shandakov, S. D., Timmermans, M. Y., et al. (2011). Synthesis of single-walled carbon nanotubes by aerosol method. *Inorganic Materials: Applied Research*, 2(6), 589–595. <https://doi.org/10.1134/S2075113311060104>
92. Tian, Y., Nasibulin, A. G., Aitchison, B., et al. (2011). Controlled synthesis of single-walled carbon nanotubes in an aerosol reactor. *The Journal of Physical Chemistry C*, 115(15), 7309–7318. <https://doi.org/10.1021/jp112291f>
93. Sztucki, M., & Narayanan, T. (2007). Development of an ultra-small-angle X-ray scattering instrument for probing the microstructure and the dynamics of soft matter. *Journal of Applied Crystallography*, 40(s1), s459–s462. <https://doi.org/10.1107/S0021889806045833>
94. Narayanan, T., Sztucki, M., Van Vaerenbergh, P., et al. (2018). A multipurpose instrument for time-resolved ultra-small-angle and coherent X-ray scattering. *Journal of Applied Crystallography*, 51(6), 1511–1524. <https://doi.org/10.1107/S1600576718012748>
95. Bardeen, J., Cooper, L. N., & Schrieffer, J. R. (1957). Theory of superconductivity. *Physical Review*, 108(5), 1175–1204. <https://doi.org/10.1103/PhysRev.108.1175>
96. Tsuei, C. C., & Kirtley, J. R. (2000). Pairing symmetry in cuprate superconductors. *Reviews of Modern Physics*, 72(4), 969–1014. <https://doi.org/10.1103/RevModPhys.72.969>
97. Ma, J., Quitmann, C., Kelley, R., et al. (1995). Temperature dependence of the superconducting gap anisotropy in $\text{Bi}_2\text{Sr}_2\text{CaCu}_2\text{O}_{8+x}$. *Science*, 267(5199), 862–865. <https://doi.org/10.1126/science.267.5199.862>

98. Zyuzin, A., Alidoust, M., & Loss, D. (2016). Josephson junction through a disordered topological insulator with helical magnetization. *Physical Review B*, 93(21), 214502. <https://doi.org/10.1103/PhysRevB.93.214502>
99. Nie, Y., & Coffey, L. (1999). Elastic and inelastic quasiparticle tunneling between anisotropic superconductors. *Physical Review B*, 59(18), 11982–11992. <https://doi.org/10.1103/PhysRevB.59.11982>
100. Wei, J. Y. T., Tsuei, C. C., van Bentum, P. J. M., et al. (1998). Quasiparticle tunneling spectra of the high-T_c mercury cuprates: Implications of the d-wave two-dimensional van Hove scenario. *Physical Review B*, 57(6), 3650–3653. <https://doi.org/10.1103/PhysRevB.57.3650>
101. Wei, J. Y. T., Yeh, N.-C., Garrigus, D. F., & Strasik, M. (1998). Directional tunneling and Andreev reflection on YBa₂Cu₃O_{7-δ} single crystals: Predominance of d-wave pairing symmetry verified with the generalized Blonder, Tinkham, and Klapwijk theory. *Physical Review Letters*, 81(12), 2542. <https://doi.org/10.1103/PhysRevLett.81.2542>
102. Sun, Y., Lv, J., Xie, Y., et al. (2019). Route to a superconducting phase above room temperature in electron-doped hydride compounds under high pressure. *Physical Review Letters*, 123(9), 097001. <https://doi.org/10.1103/PhysRevLett.123.097001>
103. Dzero, M., Xia, J., Galitski, V., & Coleman, P. (2016). Topological Kondo insulators. *Annual Review of Condensed Matter Physics*, 7, 249–280. <https://doi.org/10.1146/annurev-conmatphys-031214-014749>
104. Fu, L., & Kane, C. L. (2007). Topological insulators with inversion symmetry. *Physical Review B*, 76(4), 045302. <https://doi.org/10.1103/PhysRevB.76.045302>
105. Lenoir, B., Cassart, M., Michenaud, J.-P., Scherrer, H., & Scherrer, S. (1996). Transport properties of Bi-rich Bi-Sb alloys. *Journal of Physics and Chemistry of Solids*, 57(1), 89–99. [https://doi.org/10.1016/0022-3697\(95\)00148-4](https://doi.org/10.1016/0022-3697(95)00148-4)
106. Lenoir, B., Dauscher, A., Devaux, X., et al. (1996). Bi-Sb alloys: An update. In *Proceedings of the 1996 15th International Conference on Thermoelectrics, ICT '96* (pp. 1–13). IEEE. <https://doi.org/10.1109/ICT.1996.553246>
107. Wyss, K. M., Beckham, J. L., Chen, W., et al. (2021). Converting plastic waste pyrolysis ash into flash graphene. *Carbon*, 174, 430–438. <https://doi.org/10.1016/j.carbon.2020.12.063>
108. Coldea, A. I. (2010). Quantum oscillations probe the normal electronic states of novel superconductors. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1924), 3503–3517. <https://doi.org/10.1098/rsta.2010.0089>
109. Blaha, P., Schwarz, K., Tran, F., et al. (2020). WIEN2k: An APW+lo program for calculating the properties of solids. *The Journal of Chemical Physics*, 152(7), 074101. <https://doi.org/10.1063/1.5143061>
110. Wang, P., Yu, G., Jia, Y., et al. (2021). Landau quantization and highly mobile fermions in an insulator. *Nature*, 589, 220–224. <https://doi.org/10.1038/s41586-020-03084-9>
111. Keçeci, M. (2011). 2n-dimensional Fujii model instanton-like solutions and coupling constant's role between instantons with higher derivatives. *Turkish Journal of Physics*, 35(2), 173–178. <https://doi.org/10.3906/fiz-1012-66>

112. Du, J., Xu, N., Peng, X., et al. (2010). NMR implementation of a molecular hydrogen quantum simulation with adiabatic state preparation. *Physical Review Letters*, 104(3), 030502. <https://doi.org/10.1103/PhysRevLett.104.030502>
113. Albash, T., & Lidar, D. A. (2018). Adiabatic quantum computation. *Reviews of Modern Physics*, 90(1), 015002. <https://doi.org/10.1103/RevModPhys.90.015002>
114. Davis, D., Dods, V., Traub, C., & Yang, J. (2017). Geodesics on the regular tetrahedron and the cube. *Discrete Mathematics*, 340(1), 3183–3196. <https://doi.org/10.1016/j.disc.2016.07.004>
115. Fuchs, D. (2021). Billiard trajectories in regular polygons and geodesics on regular polyhedra. *Arnold Mathematical Journal*, 7, 493–517. <https://doi.org/10.1007/s40598-020-00170-8>
116. Athreya, J. S., Aulicino, D., Hooper, W. P., & Randecker, A. (2020). Platonic solids and high genus covers of lattice surfaces. *Experimental Mathematics*, 31(3), 847–877. <https://doi.org/10.1080/10586458.2020.1712564>
117. Zheng, Y.-C., & Brun, T. A. (2012). Geometric manipulation of ensembles of atoms on an atom chip for quantum computation. *Physical Review A*, 86(3), 032323. <https://doi.org/10.1103/PhysRevA.86.032323>
118. Hasan, M. Z., Hsieh, D., Xia, Y., et al. (2011). A new experimental approach for the exploration of topological quantum phenomena: Topological insulators and superconductors. [arXiv:1105.0396] <https://doi.org/10.48550/arXiv.1105.0396>
119. Keçeci, M. (2020). Discourse on the second quantum revolution and nanotechnology applications in the midst of the COVID-19 pandemic of inequality. *International Journal of Latest Research in Science and Technology*, 9(5), 1–7. <https://doi.org/10.5281/zenodo.7483395>; https://www.mnkjournals.com/journal/ijlrst/Article.php?paper_id=11004
120. Keçeci, M. (2001). Konformal spinör alan teorileri [Conformal spinor field theories] [Master's thesis, Gebze Technical University]. YÖK National Thesis Center. <https://tez.yok.gov.tr/UlusalTezMerkezi/tezSorguSonucYeni.jsp> (Thesis No: 109951)
121. Kadison, R. V., & Singer, I. M. (1959). Extensions of pure states. *American Journal of Mathematics*, 81(2), 383–400. <https://doi.org/10.2307/2372748>
122. Casazza, P. G., & Tremain, J. C. (2006). The Kadison-Singer problem in mathematics and engineering. *Proceedings of the National Academy of Sciences*, 103(7), 2032–2039. <https://doi.org/10.1073/pnas.0507888103>
123. Anderson, J. (1979). Extensions, restrictions, and representations of states on C-algebras. *Transactions of the American Mathematical Society*, 249, 303–329. <https://doi.org/10.1090/S0002-9947-1979-0525675-1>
124. Marcus, A., Spielman, D. A., & Srivastava, N. (2014). Interlacing families II: Mixed characteristic polynomials and the Kadison-Singer problem. [arXiv:1306.3969] <https://doi.org/10.48550/arXiv.1306.3969>
125. Casazza, P. G., & Tremain, J. C. (2015). Consequences of the Marcus/Spielman/Srivastava solution to the Kadison-Singer problem. [arXiv:1407.4768] <https://doi.org/10.48550/arXiv.1407.4768>
126. Vepsäläinen, A. P., Karamlou, A. H., et al. (2020). Impact of ionizing radiation on superconducting qubit coherence. *Nature*, 584, 551–556. <https://doi.org/10.1038/s41586-020-2619-8>

127. Cardani, L., Valenti, F., et al. (2021). Reducing the impact of radioactivity on quantum circuits in a deep-underground facility. *Nature Communications*, 12(2733). <https://doi.org/10.1038/s41467-021-23032-z>
128. DiVincenzo, D. P. (2000). The physical implementation of quantum computation. *Fortschritte der Physik*, 48(9–11), 771–783. [https://doi.org/10.1002/1521-3978\(200009\)48:9/11%3C771::AID-PROP771%3E3.0.CO;2-E](https://doi.org/10.1002/1521-3978(200009)48:9/11%3C771::AID-PROP771%3E3.0.CO;2-E)
129. Arute, F., Arya, K., Babbush, R., et al. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574, 505–510. <https://doi.org/10.1038/s41586-019-1666-5>
130. Kandala, A., Temme, K., et al. (2019). Error mitigation extends the computational reach of a noisy quantum processor. *Nature*, 567, 491–495. <https://doi.org/10.1038/s41586-019-1040-7>
131. Lutchyn, R. M., Glazman, L. I., & Larkin, A. I. (2006). Kinetics of the superconducting charge qubit in the presence of a quasiparticle. *Physical Review B*, 74(6), 064515; Erratum, 75(22), 229903(E). <https://doi.org/10.1103/PhysRevB.74.064515>, <https://doi.org/10.1103/PhysRevB.75.229903>
132. Martinis, J. M., Ansmann, M., & Aumentado, J. (2009). Energy decay in superconducting Josephson-junction qubits from nonequilibrium quasiparticle excitations. *Physical Review Letters*, 103(9), 097002. <https://doi.org/10.1103/PhysRevLett.103.097002>
133. Jin, X., Kamal, A., Sears, A. P., et al. (2015). Thermal and residual excited-state population in a 3D transmon qubit. *Physical Review Letters*, 114(24), 240501. <https://doi.org/10.1103/PhysRevLett.114.240501>
134. Serniak, K., Hays, M., de Lange, G., et al. (2018). Hot nonequilibrium quasiparticles in transmon qubits. *Physical Review Letters*, 121(15), 157701. <https://doi.org/10.1103/PhysRevLett.121.157701>
135. Aumentado, J., Keller, M. W., Martinis, J. M., & Devoret, M. H. (2004). Nonequilibrium quasiparticles and 2e periodicity in single-Cooper-pair transistors. *Physical Review Letters*, 92(6), 066802. <https://doi.org/10.1103/PhysRevLett.92.066802>
136. Taupin, M., Khaymovich, I., Meschke, M., et al. (2016). Tunable quasiparticle trapping in Meissner and vortex states of mesoscopic superconductors. *Nature Communications*, 7(10977). <https://doi.org/10.1038/ncomms10977>
137. Serniak, K., Diamond, S., Hays, M., et al. (2019). Direct dispersive monitoring of charge parity in offset-charge-sensitive transmons. *Physical Review Applied*, 12(1), 014052. <https://doi.org/10.1103/PhysRevApplied.12.014052>
138. Córcoles, A. D., Chow, J. M., Gambetta, J. M., et al. (2011). Protecting superconducting qubits from radiation. *Applied Physics Letters*, 99(18), 181906. <https://doi.org/10.1063/1.3658630>
139. Barends, R., Wenner, J., Lenander, M., et al. (2011). Minimizing quasiparticle generation from stray infrared light in superconducting quantum circuits. *Applied Physics Letters*, 99(11), 113507. <https://doi.org/10.1063/1.3638063>
140. Bepalov, A., Houzet, M., Meyer, J. S., & Nazarov, Y. V. (2016). Theoretical model to explain excess of quasiparticles in superconductors. *Physical Review Letters*, 117(11), 117002. <https://doi.org/10.1103/PhysRevLett.117.117002>
141. Dicke, R. H. (1946). The measurement of thermal radiation at microwave frequencies. *Review of Scientific Instruments*, 17, 268–275. <https://doi.org/10.1063/1.1770483>

142. Agnese, R., et al. (2017). Projected sensitivity of the SuperCDMS SNOLAB experiment. *Physical Review D*, 95(8), 082002. <https://doi.org/10.1103/PhysRevD.95.082002>
143. Alduino, C., et al. (2018). First results from CUORE: A search for lepton number violation via $0\nu\beta\beta$ decay of ^{130}Te . *Physical Review Letters*, 120(13), 132501. <https://doi.org/10.1103/PhysRevLett.120.132501>
144. Agostini, M., et al. (2018). Improved limit on neutrinoless double- β decay of ^{76}Ge from GERDA phase II. *Physical Review Letters*, 120(13), 132503. <https://doi.org/10.1103/PhysRevLett.120.132503>
145. Gando, A., et al. (2016). Search for Majorana neutrinos near the inverted mass hierarchy region with KamLAND-Zen. *Physical Review Letters*, 117(8), 082503. <https://doi.org/10.1103/PhysRevLett.117.082503>
146. Aalseth, C. E., et al. (2018). Search for neutrinoless double- β decay in ^{76}Ge with the Majorana demonstrator. *Physical Review Letters*, 120(13), 132502. <https://doi.org/10.1103/PhysRevLett.120.132502>
147. Albert, J. B., et al. (2018). Search for neutrinoless double-beta decay with the upgraded EXO-200 detector. *Physical Review Letters*, 120(7), 072701. <https://doi.org/10.1103/PhysRevLett.120.072701>
148. Erhard, A., Poulsen Nautrup, H., Meth, M., et al. (2021). Entangling logical qubits with lattice surgery. *Nature*, 589, 220–224. <https://doi.org/10.1038/s41586-020-03079-6>
149. Ghosh, S., Shekhter, A., Jerzembeck, F., et al. (2020). Thermodynamic evidence for a two-component superconducting order parameter in Sr_2RuO_4 . *Nature Physics*, 17, 199–204. [arXiv:2002.06130v2] <https://doi.org/10.1038/s41567-020-1032-4>
150. Stanescu, T. D., Galitski, V., Vaishnav, J. Y., Clark, C. W., & Das Sarma, S. (2009). *Physical Review A*, 79(5), 053639. [arXiv:0901.3921v1] <https://doi.org/10.1103/PhysRevA.79.053639>
151. Yu, P., Chen, J., Gomanko, M., et al. (2021). Non-Majorana states yield nearly quantized conductance in proximatized nanowires. *Nature Physics*, 17(4), 482–488. <https://doi.org/10.1038/s41567-020-01107-w>
152. Chen, Z.-Y., Zhou, Q., Xue, C., et al. (2018). 64-qubit quantum circuit simulation. *Science Bulletin*, 63(15), 964–971. <https://doi.org/10.1016/j.scib.2018.06.007>
153. Wang, Z., Chen, Z., Wang, S., et al. (2021). A quantum circuit simulator and its applications on Sunway TaihuLight supercomputer. *Scientific Reports*, 11(1), 355. <https://doi.org/10.1038/s41598-020-79777-y>
154. Zhang, H., & Ding, F. (2013). On the Kronecker products and their applications. *Journal of Applied Mathematics*, 2013, 296185. <https://doi.org/10.1155/2013/296185>
155. Marques, J. F., Varbanov, B. M., Moreira, M. S., et al. (2021). Logical-qubit operations in an error-detecting surface code. *Nature Physics*, 18(1), 80–86. <https://doi.org/10.1038/s41567-021-01423-9>
156. Versluis, R., Poletto, S., Khammassi, N., et al. (2017). Scalable quantum circuit and control for a superconducting surface code. *Physical Review Applied*, 8(3), 034021. <https://doi.org/10.1103/PhysRevApplied.8.034021>

157. Fowler, A. G., Mariantoni, M., Martinis, J. M., & Cleland, A. N. (2012). Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3), 032324. <https://doi.org/10.1103/PhysRevA.86.032324>
158. Jones, N. C., Meter, R. V., Fowler, A. G., et al. (2012). Layered architecture for quantum computing. *Physical Review X*, 2(3), 031007. <https://doi.org/10.1103/PhysRevX.2.031007>
159. Bravyi, S. B., & Kitaev, A. Y. (1998). Quantum codes on a lattice with boundary. [arXiv:quant-ph/9811052] <https://doi.org/10.48550/arXiv.quant-ph/9811052>
160. Harrow, A. W., Hassidim, A., & Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15), 150502. <https://doi.org/10.1103/PhysRevLett.103.150502>
161. Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5), 1484–1509. <https://doi.org/10.1137/S0097539795293172>
162. Church, A. (1936). An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2), 345–363. <https://doi.org/10.2307/2268571>
163. Turing, A. M. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1), 230–265. <https://doi.org/10.1112/plms/s2-42.1.230>; *Proceedings of the London Mathematical Society*, s2-43(1938), 544–546. <https://doi.org/10.1112/plms/s2-43.6.544>
164. Post, E. L. (1936). Finite combinatory processes—formulation 1. *The Journal of Symbolic Logic*, 1(3), 103–105. <https://doi.org/10.2307/2269031>
165. Hartmanis, J., & Simon, J. (1974). On the power of multiplication in random access machines. In *Proceedings of the 15th Annual Symposium on Switching and Automata Theory* (pp. 13–23). IEEE Computer Society. <https://doi.org/10.1109/SWAT.1974.20>
166. Vergis, A., Steiglitz, K., & Dickinson, B. (1986). The complexity of analog computation. *Mathematics and Computers in Simulation*, 28(2), 91–113. [https://doi.org/10.1016/0378-4754\(86\)90105-9](https://doi.org/10.1016/0378-4754(86)90105-9)
167. Toffoli, T. (1980). Reversible computing. In *International Colloquium on Automata, Languages and Programming* (pp. 632–644). Springer. *Lecture Notes in Computer Science*, Vol. 85. https://doi.org/10.1007/3-540-10003-2_104
168. Fredkin, E., & Toffoli, T. (1982). Conservative logic. *International Journal of Theoretical Physics*, 21(3/4), 219–253. <https://doi.org/10.1007/BF01857727>
169. Dawson, C. M., & Nielsen, M. A. (2006). The Solovay-Kitaev algorithm. *Quantum Information & Computation*, 6(1), 81–95. [arXiv:quant-ph/0505030] <https://doi.org/10.48550/arXiv.quant-ph/0505030>
170. Kitaev, A. Y. (1997). Quantum computations: Algorithms and error correction. *Russian Mathematical Surveys*, 52(6), 1191–1249. <https://doi.org/10.1070/RM1997v052n06ABEH002155>
171. Williams, C. P. (2011). Quantum gates. In *Explorations in quantum computing* (pp. 51–122). Springer. https://doi.org/10.1007/978-1-84628-887-6_2
172. Deutsch, D. E. (1989). Quantum computational networks. *Proceedings of the Royal Society A*, 425(1868), 73–90. <https://doi.org/10.1098/rspa.1989.0099>

173. Fowler, A. G., Mariantoni, M., Martinis, J. M., & Cleland, A. N. (2012). Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3), 032324. <https://doi.org/10.1103/PhysRevA.86.032324>
174. Fowler, A. G. (2013). Polyestimate: Instantaneous open source surface code analysis. [arXiv:1307.0689] <https://doi.org/10.48550/arXiv.1307.0689>
175. Modi, K. (2009). A theoretical analysis of experimental open quantum dynamics. [arXiv:0903.2724] <https://doi.org/10.48550/arXiv.0903.2724>
176. Korotkov, A. N. (2013). Error matrices in quantum process tomography. [arXiv:1309.6405] <https://doi.org/10.48550/arXiv.1309.6405>
177. Baldwin, C. H., Kalev, A., & Deutsch, I. H. (2014). Quantum process tomography of unitary and near-unitary maps. *Physical Review A*, 90(1), 012110. [arXiv:1404.2877] <https://doi.org/10.1103/PhysRevA.90.012110>
178. Kubica, A., Yoshida, B., & Pastawski, F. (2015). Unfolding the color code. *New Journal of Physics*, 17(8), 083026. <https://doi.org/10.1088/1367-2630/17/8/083026>
179. Ghosh, J., Fowler, A. G., & Geller, M. R. (2012). Surface code with decoherence: An analysis of three superconducting architectures. *Physical Review A*, 86(6), 062318. [arXiv:1210.5799] <https://doi.org/10.1103/PhysRevA.86.062318>
180. Bullock, S. S., & Brennen, G. K. (2007). Qudit surface codes and gauge theory with finite cyclic groups. *Journal of Physics A: Mathematical and Theoretical*, 40(13), 3481–3505. [arXiv:quant-ph/0609070] <https://doi.org/10.1088/1751-8113/40/13/013>
181. Levin, M. A., & Wen, X.-G. (2005). String-net condensation: A physical mechanism for topological phases. *Physical Review B*, 71(4), 045110. [arXiv:cond-mat/0404617] <https://doi.org/10.1103/PhysRevB.71.045110>
182. Wootton, J. R., Lahtinen, V., Doucot, B., & Pachos, J. K. (2011). Engineering complex topological memories from simple Abelian models. *Annals of Physics*, 326(9), 2307–2314. [arXiv:0908.0708] <https://doi.org/10.1016/j.aop.2011.05.008>
183. Kitaev, A. Yu. (2003). Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1), 2–30. [arXiv:quant-ph/9707021] [https://doi.org/10.1016/S0003-4916\(02\)00018-0](https://doi.org/10.1016/S0003-4916(02)00018-0)
184. Aaronson, S., Grier, D., & Schaeffer, L. (2015). The classification of reversible bit operations. [arXiv:1504.05155] <https://doi.org/10.48550/arXiv.1504.05155>
185. Huang, H.-Y. (2022). Learning quantum states from classical shadows. *Nature Reviews Physics*, 4(2), 81–82. <https://doi.org/10.1038/s42254-021-00411-5>
186. Born, M. (1926). Zur Quantenmechanik der Stoßvorgänge. *Zeitschrift für Physik*, 37(12), 863–867. <https://doi.org/10.1007/BF01397477>
187. Kitaev, A. Yu. (1997). Quantum computing: algorithms and error correction. *Russian Math. Surveys* 6. <https://doi.org/10.1070/RM1997v052n06ABEH002155>
188. Kitaev, A. Yu. (1997). Quantum error correction with imperfect gates. In *Quantum Communication, Computing, and Measurement* (pp. 181–188). Springer. https://doi.org/10.1007/978-1-4615-5923-8_19
189. Bombin, H., & Martin-Delgado, M. A. (2006). Topological quantum distillation. *Physical Review Letters*, 97(18), 180501. [arXiv:quant-ph/0605138] <https://doi.org/10.1103/PhysRevLett.97.180501>

190. Freedman, M., & Meyer, D. (2001). Projective plane and planar quantum codes. *Foundations of Computational Mathematics*, 1(3), 325–332. [arXiv:quant-ph/9810055]
<https://doi.org/10.1007/s102080010013>
191. Kitaev, A. Yu., & Laumann, C. (2009). Topological phases and quantum computation. [arXiv:0904.2771] <https://doi.org/10.48550/arXiv.0904.2771>
192. Nanda, A., Dhochak, K., & Bhattacharjee, S. (2020). Phases and quantum phase transitions in an anisotropic ferromagnetic Kitaev-Heisenberg- Γ magnet. *Physical Review B*, 102(23), 235124. [arXiv:2006.10081] <https://doi.org/10.1103/PhysRevB.102.235124>
193. Gokhale, P., Baker, J. M., Duckering, C., et al. (2019). Asymptotic improvements to quantum circuits via qutrits. In *ISCA '19: Proceedings of the 46th International Symposium on Computer Architecture* (pp. 554–566). ACM. <https://doi.org/10.1145/3307650.3322253>
194. Das, P., Pattison, C. A., Manne, S., et al. (2020). A scalable decoder micro-architecture for fault-tolerant quantum computing. [arXiv:2001.06598]. <https://doi.org/10.48550/arXiv.2001.06598>
195. Hu, M. S. (2020). Quasilinear time decoding algorithm for topological codes with high error threshold [Technical Report]. ResearchGate. <https://doi.org/10.13140/RG.2.2.13495.96162>
196. Kolmogorov, V. (2009). Blossom V: A new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1), 43–67. <https://doi.org/10.1007/s12532-009-0002-8>
197. Varsamopoulos, S., Bertels, K., & Almudever, C. G. (2020). Decoding surface code with a distributed neural network-based decoder. *Quantum Machine Intelligence*, 2(3), 1–10.
<https://doi.org/10.1007/s42484-020-00015-9>
198. Ollivier, H., & Tillich, J.-P. (2006). Trellises for stabilizer codes: Definition and uses. *Physical Review A*, 74(3), 032304. <https://doi.org/10.1103/PhysRevA.74.032304>
199. Feynman, R. P. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6–7), 467–488. <https://doi.org/10.1007/BF02650179>
200. <https://docs.quantum.ibm.com/migration-guides/local-simulators>; https://qiskit.github.io/qiskit-aer/tutorials/6_extended_stabilizer_tutorial.html
201. https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.StatevectorSimulator.html;
https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.UnitarySimulator.html
202. https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.AerSimulator.html; https://qiskit.github.io/qiskit-aer/tutorials/7_matrix_product_state_method.html
203. https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.QasmSimulator.html
204. <https://learn.microsoft.com/en-us/azure/quantum/backend-simulators>; <https://learn.microsoft.com/en-us/azure/quantum/sparse-simulator>
205. <https://learn.microsoft.com/tr-tr/azure/quantum>
206. Low, G. H., Bauman, N. P., Granade, C. E., et al. (2019). Q# and NWChem: Tools for scalable quantum chemistry on quantum computers. [arXiv:1904.01131]
<https://doi.org/10.48550/arXiv.1904.01131>
207. Low, G. H., Munro, W. J. (2015). Optimal Trotterization in universal quantum simulators under faulty control. *Physical Review A*, 91(3), 052327. [arXiv:1502.04536]
<https://doi.org/10.1103/PhysRevA.91.052327>

208. Bravyi, S., Suchara, M., & Vargo, A. (2014). Efficient algorithms for maximum likelihood decoding in the surface code. *Physical Review A*, 90(3), 032326. <https://doi.org/10.1103/PhysRevA.90.032326>
209. Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2), 147–160. <https://doi.org/10.1002/j.1538-7305.1950.tb00463.x>
210. Lin, S. (1970). *An introduction to error-correcting codes*. Prentice-Hall.
<https://doi.org/10.1109/TIT.1971.1054715>
211. MacWilliams, F. J., & Sloane, N. J. A. (1977). *The theory of error-correcting codes*. North-Holland. ISBN 978-0444850102
212. Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3), 379–423. <https://doi.org/10.1002/j.1538-7305.1948.tb01338.x>
213. Fowler, A. G. (2013). Optimal complexity correction of correlated errors in the surface code. [arXiv:1310.0863] <https://doi.org/10.48550/arXiv.1310.0863>
214. Fowler, A. G., Stephens, A. M., & Groszkowski, P. (2013). High-threshold universal quantum computation with the surface code. *Physical Review A*, 80(5), 052312. [arXiv:0803.0272] <https://doi.org/10.1103/PhysRevA.80.052312>
215. Duclos-Cianci, G., & Poulin, D. (2010). Fast decoders for topological quantum codes. *Physical Review Letters*, 104(5), 050504. <https://doi.org/10.1103/PhysRevLett.104.050504>
216. Torlai, G., & Melko, R. G. (2017). Neural decoder for topological codes. *Physical Review Letters*, 119(3), 030501. <https://doi.org/10.1103/PhysRevLett.119.030501>
217. Varsamopoulos, S., Criger, B., & Bertels, K. (2018). Decoding small surface codes with feedforward neural networks. *Quantum Science and Technology*, 3(1), 015004. <https://doi.org/10.1088/2058-9565/aa955a>
218. Atiyah, M. F., & Anderson, D. W. (1967). *K-theory*. W. A. Benjamin. 2nd ed. (1989). ISBN 9780201093940.
219. Bass, H. (1968). *Algebraic K-theory*. W. A. Benjamin.
220. Atiyah, M., & Segal, G. (2004). Twisted K-theory. *Ukrainian Mathematical Bulletin*, 1(3), 287–330. [arXiv:math/0407054] <https://doi.org/10.48550/arXiv.math/0407054>
221. Atiyah, M. F., & Segal, G. (2006). Twisted K-theory and cohomology. *Nankai Tracts in Mathematics*, 11, 5–43. [arXiv:math/0510674] <https://doi.org/10.48550/arXiv.math/0510674>
222. Conner, P. E., & Floyd, E. E. (1964). *Differentiable periodic maps*. Springer. *Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 33*. <https://doi.org/10.1007/978-3-662-41633-4>
223. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. [arXiv:1412.6980] <https://doi.org/10.48550/arXiv.1412.6980>
224. Reddi, S. J., Kale, S., & Kumar, S. (2018). On the convergence of Adam and beyond. [arXiv:1904.09237] <https://doi.org/10.48550/arXiv.1904.09237>
225. Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning* (Vol. 28, pp. 1139–1147). JMLR. <https://proceedings.mlr.press/v28/sutskever13.html>
226. McMahan, B., Holt, G., Sculley, D., et al. (2013). Ad click prediction: A view from the trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1222–1230). ACM. <https://doi.org/10.1145/2487575.2488200>

227. Kottmann, K., Metz, F., Fraxanet, J., & Baldelli, N. (2021). Variational quantum anomaly detection: Unsupervised mapping of phase diagrams on a physical quantum computer. *Physical Review Research*, 3(4), 043184. [arXiv:2106.07912] <https://doi.org/10.1103/PhysRevResearch.3.043184>
228. Ashikhmin, A., & Knill, E. (2000). Nonbinary quantum stabilizer codes. [arXiv:quant-ph/0005008] <https://doi.org/10.48550/arXiv.quant-ph/0005008>
229. Klimov, A. B., Muñoz, C., & Sánchez-Soto, L. L. (2009). Discrete coherent and squeezed states of many-qudit systems. *Physical Review A*, 80(4), 043836. <https://doi.org/10.1103/PhysRevA.80.043836>
230. Lidl, R., & Niederreiter, H. (1994). Introduction to finite fields and their applications (2nd ed.). Cambridge University Press. <https://doi.org/10.1017/CBO9781139172769>
231. Knill, E., Laflamme, R., & Viola, L. (2000). Theory of quantum error correction for general noise. *Physical Review Letters*, 84(11), 2525–2528. [arXiv:quant-ph/9604034] <https://doi.org/10.1103/PhysRevLett.84.2525>
232. Bennett, C. H., DiVincenzo, D. P., Smolin, J. A., & Wootters, W. K. (1996). Mixed-state entanglement and quantum error correction. *Physical Review A*, 54(5), 3824–3851. [arXiv:quant-ph/9604024] <https://doi.org/10.1103/PhysRevA.54.3824>
233. Li, Y., & Benjamin, S. C. (2017). Efficient variational quantum simulator incorporating active error minimization. *Physical Review X*, 7(2), 021050. <https://doi.org/10.1103/PhysRevX.7.021050>
234. Temme, K., Bravyi, S., & Gambetta, J. M. (2017). Error mitigation for short-depth quantum circuits. *Physical Review Letters*, 119(18), 180509. <https://doi.org/10.1103/PhysRevLett.119.180509>
235. McClean, J. R., Kimchi-Schwartz, M. E., Carter, J., & de Jong, W. A. (2017). Hybrid quantum-classical hierarchy for mitigation of decoherence and determination of excited states. *Physical Review A*, 95(4), 042308. <https://doi.org/10.1103/PhysRevA.95.042308>
236. Edmonds, J. (1965). Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards-B*, 69B, 125–130. <https://doi.org/10.6028/jres.069B.013>
237. Edmonds, J. (1965). Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17, 449–467. <https://doi.org/10.4153/CJM-1965-045-4>
238. Romero, R. (2015). Qubits, Weyl spinors, quantum NOT gates, and dynamical decoupling. [arXiv:1412.1158] <https://doi.org/10.48550/arXiv.1412.1158>
239. Ukhtary, M. S., Hasdeo, E. H., Suksmono, A. B., & Nugraha, A. R. T. (2022). Long-lived qubit entanglement by surface plasmon polaritons in a Weyl semimetal. *Physical Review B*, 106(15), 155409. [arXiv:2206.00534] <https://doi.org/10.1103/PhysRevB.106.155409>
240. Chiu, K.-L., Qian, D., & others. (2020). Flux-tunable superconducting quantum circuit based on Weyl semimetal MoTe₂. *Nano Letters*, 20(12), 8469–8475. [arXiv:2010.14107] <https://doi.org/10.1021/acs.nanolett.0c02267>
241. Mari, A., Shammah, N., & Zeng, W. J. (2021). Extending quantum probabilistic error cancellation by noise scaling. *Physical Review A*, 104(5), 052607. <https://doi.org/10.1103/PhysRevA.104.052607>
242. Sun, J., Yuan, X., Tsunoda, T., et al. (2021). Mitigating realistic noise in practical NISQ devices. *Physical Review Applied*, 15(3), 034026. <https://doi.org/10.1103/PhysRevApplied.15.034026>

243. Zhang, S., Lu, Y., Zhang, K., et al. (2020). Error-mitigated quantum gates exceeding physical fidelities in a trapped-ion system. *Nature Communications*, 11(1), 587. <https://doi.org/10.1038/s41467-020-14376-z>
244. Endo, S., Benjamin, S. C., & Li, Y. (2018). Practical quantum error mitigation for near-future applications. *Physical Review X*, 8(3), 031027. <https://doi.org/10.1103/PhysRevX.8.031027>
245. Song, C., Cui, J., Wang, H., et al. (2019). Quantum computation with universal error mitigation on a superconducting quantum processor. *Science Advances*, 5(9), eaaw5686. <https://doi.org/10.1126/sciadv.aaw5686>
246. Berg, E., Mineev, Z. K., Kandala, A., & Temme, K. (2022). Probabilistic error cancellation with sparse Pauli-Lindblad models on noisy quantum processors. [arXiv:2201.09866] <https://doi.org/10.48550/arXiv.2201.09866>
247. Berg, E., Mineev, Z. K., & Temme, K. (2022). Model-free readout-error mitigation for quantum expectation values. *Physical Review A*, 105(3), 032620. [arXiv:2012.09738] <https://doi.org/10.1103/PhysRevA.105.032620>
248. Nielsen, M. A., & Chuang, I. L. (2010). *Quantum computation and quantum information* (10th anniversary ed.). Cambridge University Press. <https://doi.org/10.1017/CBO9780511976667>
249. Czarnik, P., Arrasmith, A., Coles, P. J., & Cincio, L. (2021). Error mitigation with Clifford quantum-circuit data. *Quantum*, 5, 592. <https://doi.org/10.22331/q-2021-11-26-592>
250. Cross, A. W., Bishop, L. S., Sheldon, S., Nation, P. D., & Gambetta, J. M. (2019). Validating quantum computers using randomized model circuits. *Physical Review A*, 100(3), 032328. [arXiv:1811.02262] <https://doi.org/10.1103/PhysRevA.100.032328>
251. Kuroiwa, K., & Nakagawa, Y. O. (2023). Clifford+T-gate decomposition with limited T gates, its error analysis, and performance of UCC ansatz in pre-FTQC era. [arXiv:2301.04150] <https://doi.org/10.48550/arXiv.2301.04150>
252. Coppersmith, D., & Winograd, S. (1990). Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3), 251–280. [https://doi.org/10.1016/S0747-7171\(08\)80013-2](https://doi.org/10.1016/S0747-7171(08)80013-2)
253. Williams, V. V. (2012). Multiplying matrices faster than Coppersmith-Winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing* (pp. 887–898). ACM. <https://doi.org/10.1145/2213955.2214056>
254. Le Gall, F. (2014). Algebraic complexity theory and matrix multiplication. In *ISSAC '14: Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation* (p. 23). ACM. [arXiv:1401.7714] <https://doi.org/10.1145/2608628.2627493>
255. Alman, J., & Williams, V. V. (2020). A refined laser method and faster matrix multiplication. [arXiv:2010.05846] <https://doi.org/10.48550/arXiv.2010.05846>
256. Duan, R., Wu, H., & Zhou, R. (2022). Faster matrix multiplication via asymmetric hashing. [arXiv:2210.10173] <https://doi.org/10.48550/arXiv.2210.10173>
257. Zhou, W., Miura, A., Sakuraba, Y., et al. (2023). Direct electrical probing of anomalous Nernst conductivity. [arXiv:2301.02465] <https://doi.org/10.48550/arXiv.2301.02465>
258. Panchenko, M., Auler, R., Nell, B., & Ottoni, G. (2018). BOLT: A practical binary optimizer for data centers and beyond. [arXiv:1807.06735] <https://doi.org/10.48550/arXiv.1807.06735>

259. Fan, X., Soin, N., Li, H., et al. (2020). Fullerene (C_{60}) nanowires: Preparation, characterization, and potential applications. *Energy & Environmental Materials*, 3(4), 469–491. <https://doi.org/10.1002/eem2.12071>
260. Umeta, Y., Suga, H., Takeuchi, M., et al. (2021). C_{60} -nanowire two-state resistance switching based on fullerene polymerization/depolymerization. *ACS Applied Nano Materials*, 4(1), 820–829. <https://doi.org/10.1021/acsanm.0c03144>
261. Suzuki, M. (1982). The μ -number constant stratum of a quasihomogeneous function of corank two. *Singularities in Complex Analytic Geometry*. Kyoto University. <https://www.kurims.kyoto-u.ac.jp/~kyodo/kokyuroku/contents/pdf/474-001.pdf>
262. Suzuki, M. (1983). The μ -number constant stratum of a quasihomogeneous function of corank two is smooth. *Proceedings of the Japan Academy, Series A*, 59(5), 188–190. <https://doi.org/10.3792/pjaa.59.188>
263. Suzuki, M. (1984). The stratum with constant Milnor number of a quasihomogeneous function of corank two. *Topology*, 23(1), 101–115. [https://doi.org/10.1016/0040-9383\(84\)90030-2](https://doi.org/10.1016/0040-9383(84)90030-2)
264. Çakıllı, H. (1977). Genel Topolojiye Giriş. İ.Ü. Fen Fakültesi Basımevi.
265. Liu, Y. (2022). Additive actions on hyperquadrics of corank two. [arXiv:2201.11268] <https://doi.org/10.48550/arXiv.2201.11268>
266. Guo, Y., Lin, Z., Zhao, J. Q., et al. (2019). Two-dimensional tunable Dirac/Weyl semimetal in non-Abelian gauge field. *Scientific Reports*, 9(1), 18516. <https://doi.org/10.1038/s41598-019-54670-5>
267. Lüscher, M. (2000). Weyl fermions on the lattice and the non-Abelian gauge anomaly. *Nuclear Physics B*, 568(1–2), 162–179. [https://doi.org/10.1016/S0550-3213\(99\)00731-2](https://doi.org/10.1016/S0550-3213(99)00731-2)
268. Guin, S. N., Vir, P., & others. (2019). Zero-field Nernst effect in a ferromagnetic Kagome-lattice Weyl-semimetal $Co_3Sn_2S_2$. *Advanced Materials*, 31(25), 1806622. <https://doi.org/10.1002/adma.201806622>
269. Guo, Z., Lu, P., Chen, T., et al. (2018). High-pressure phases of Weyl semimetals NbP, NbAs, TaP, and TaAs. *Science China Physics, Mechanics & Astronomy*, 61(3), 038211. <https://doi.org/10.1007/s11433-017-9126-6>
270. Nadj-Perge, S., Drozdov, I. K., Li, J., et al. (2014). Observation of Majorana fermions in ferromagnetic atomic chains on a superconductor. *Science*, 346(6209), 602–607. <https://doi.org/10.1126/science.1259327>
271. Zhang, G., Li, C., Song, Z. (2017). Majorana charges, winding numbers, and Chern numbers in quantum Ising models. *Scientific Reports*, 7(1), 8176. <https://doi.org/10.1038/s41598-017-08323-0>
272. Wang, D., Fu, L., et al. (2020). Signature of a pair of Majorana zero modes in superconducting gold surface states. *PNAS*, 117(16), 8775–8782. <https://doi.org/10.1073/pnas.1919753117>
273. Dongfei Wang, et al. (2018). Evidence for Majorana bound states in iron-based superconductor. *Science*, 362(6418), 333–335. <https://doi.org/10.1126/science.aao1797>
274. Litinski, D., & von Oppen, F. (2018). Quantum computing with Majorana fermion codes. *Physical Review B*, 97(20), 205404. <https://doi.org/10.1103/PhysRevB.97.205404>
275. Wong, K. H., Hirsbrunner, M. R., Gliozzi, J., et al. (2023). Higher-order topological superconductivity in magnet-superconductor hybrid systems. *npj Quantum Materials*, 8(1), 31. <https://doi.org/10.1038/s41535-023-00564-9>

276. Karzig, T., Knapp, C., Lutchyn, R. M., et al. (2017). Scalable designs for quasiparticle-poisoning-protected topological quantum computation with Majorana zero modes. *Physical Review B*, 95(23), 235305. <https://doi.org/10.1103/PhysRevB.95.235305>
277. Nam, Y., Ross, N. J., Su, Y., et al. (2018). Automated optimization of large quantum circuits with continuous parameters. *npj Quantum Information*, 4(1), 23. <https://doi.org/10.1038/s41534-018-0072-4>
278. Amy, M., Maslov, D., & Mosca, M. (2014). Polynomial-time T-depth optimization of Clifford+T circuits via matroid partitioning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(10), 1476–1489. <https://doi.org/10.1109/TCAD.2014.2341953>
279. Pikulin, D. I., Van Heck, B., & Karzig, T. (2021). Protocol to identify a topological superconducting phase in a three-terminal device. [arXiv:2103.12217] <https://doi.org/10.48550/arXiv.2103.12217>
280. Aghaee, M., Akkala, A., Alam, Z., et al. (2023). InAs-Al hybrid devices passing the topological gap protocol. *Physical Review B*, 107(24), 245423. [arXiv:2207.02472] <https://doi.org/10.1103/PhysRevB.107.245423>
281. SciPy cdist docs: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.cdist.html>
282. scikit-learn DistanceMetric docs: <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.DistanceMetric.html>
283. C++ Standards Committee Papers: <https://www.open-std.org/jtc1/sc22/wg21/docs/papers/>
284. AMD AOCC: <https://www.amd.com/en/developer/aocc.html>
285. Mirrahimi, M., Leghtas, Z., Albert, V. V., et al. (2014). Dynamically protected cat-qubits: A new paradigm for universal quantum computation. *New Journal of Physics*, 16(4), 045014. <https://doi.org/10.1088/1367-2630/16/4/045014>
286. Bergeron, H., Curado, E. M. F., Gazeau, J. P., & Rodrigues, L. M. C. (2017). A baby Majorana quantum formalism. [arXiv:1701.04026] <https://doi.org/10.48550/arXiv.1701.04026>
287. Batelaan, H., & Tonomura, A. (2009). The Aharonov–Bohm effects: Variations on a subtle theme. *Physics Today*, 62(9), 38–43. <https://doi.org/10.1063/1.3226857>
288. Aharonov, Y., & Bohm, D. (1961). Further considerations on electromagnetic potentials in the quantum theory. *Physical Review*, 123(4), 1511–1524. <https://doi.org/10.1103/PhysRev.123.1511>
289. Aksenov, S. V. (2022). Structural phase transitions and critical behavior of Fe-based superconductors. *Journal of Physics: Condensed Matter*, 34(25), 253001. <https://doi.org/10.1088/1361-648X/ac62a7>
290. Tatsuta M., Matsuzaki Y., and Shimizu A., (2019). Quantum metrology with generalized cat states. *Phys. Rev. A* 100, 032318. [arXiv: 1902.01551v2] <https://doi.org/10.1103/PhysRevA.100.032318>
291. Greenberger, D. M., Horne, M. A., Shimony, A., and Zeilinger, A. (1990). Bell’s theorem without inequalities. *American Journal of Physics* 58, 1131. <https://doi.org/10.1119%2F1.16243>
292. T. Monz, P. Schindler, J. T. Barreiro, M. Chwalla, D. Nigg, W. A. Coish, M. Harlander, W. Hänsel, M. Hennrich, and R. Blatt (2011). 14-qubit entanglement: creation and coherence. *Physical Review Letters* 106, 130506. <https://doi.org/10.1103/PhysRevLett.106.130506>

293. DiCarlo, L., Reed, M., Sun, L. et al. (2010). Preparation and measurement of three-qubit entanglement in a superconducting circuit. *Nature* 467, 574–578. <https://doi.org/10.1038/nature09416>
294. F. Fröwis, P. Sekatski, W. Dür, N. Gisin, and N. Sangouard (2018). Macroscopic quantum states: Measures, fragility, and implementations. *Reviews of Modern Physics* 90, 025004 (2018). <https://doi.org/10.1103/RevModPhys.90.025004>
295. Yaman, M., & Misir, Z. (2022). Finite-Time Behaviour of Solutions to Nonlinear Parabolic equation. *New Trends in Mathematical Sciences*, 10(4), 47–53. <https://doi.org/10.20852/ntmsci.2022.487>
296. Mustafa, O., & Güvendi, A. (2025). Fermions in a (2+1)-Dimensional Magnetized Spacetime with a Cosmological Constant: Domain Walls and Spinning Magnetic Vortices. *Physics Letters B*, Volume 866, 139569. <https://doi.org/10.1016/j.physletb.2025.139569>
297. Altun, I., Gençtürk, İ., & Erduran, A. (2023). Prešić-type fixed point results via Q-distance on quasimetric space and application to (p, q)-difference equations. *Nonlinear Analysis Modelling and Control*, 28(6), 1013-1026. <https://doi.org/10.15388/namc.2023.28.33436>
298. de M. Carvalho, A. M., Garcia, G. Q., & Furtado, C. (2025, April). Geometric and Topological Aspects of Quadrupoles of Disclinations: Conformal Metrics and Self-Forces. <https://doi.org/10.48550/arXiv.2504.21210>
299. Domuschiev, I. (2025, May). Quantum Simulation versus Model Prediction in Human Medicine. *ResearchGate*. <https://doi.org/10.13140/RG.2.2.17725.37608>
300. Yıldız, F., Przybylski, M., & Kirschner, J. (2009). Direct evidence of a nonorthogonal magnetization configuration in single crystalline Fe_{1-x}Co_x/Rh/Fe/Rh(001) system. *Physical Review Letters*, 103(14), 147203. <https://doi.org/10.1103/PhysRevLett.103.147203>
301. Mikailzade, F., Maksutoglu, M., Khaibullin, R.I., Valeev, V.F., Nuzhdin, V.I., Aliyeva, V.B., & Mammadov, T.G. (2016). Magnetodielectric Effects in Co-implanted TlInS₂ and TlGaSe₂ Crystals. *Phase Transitions*, 89(6), 568–577. <https://doi.org/10.1080/01411594.2015.1080259>
302. Yalçın, O., et al. (2023). Crystallographic, structural, optical, and dielectric properties of aniline and aniline halide imprinted hydrogels for optoelectronic applications. *Journal of Materials Science: Materials in Electronics*, 34(22), 1700. <https://doi.org/10.1007/s10854-023-10915-8>
303. Veliev, E. V., Günaydin, S., & Sundu, H. (2018). Thermal properties of the exotic X(3872) state via QCD sum rule. *The European Physical Journal Plus*, 133(3), 139. <https://doi.org/10.1140/epjp/i2018-11977-0>
304. Rameev, B. (2020). Magnetic Resonance and Microwave Techniques for Security Applications. 2019 Photonics & Electromagnetics Research Symposium-Spring (PIERS-Spring). IEEE. <https://doi.org/10.1109/PIERS-Spring46901.2019.9017563>
305. Bidai, K., Tabeti, A., Mohammed, D. S., Seddik, T., Batouche, M., Özdemir, M., & Bakhti, B. (2020). Carbon substitution enhanced electronic and optical properties of MgSiP₂ chalcopyrite through TB-mBJ approximation. *Computational Condensed Matter*, 24, e00490. <https://doi.org/10.1016/j.cocom.2020.e00490>
306. Elzwawy, A., Pişkin, H., Akdoğan, N., et al. (2021). Current trends in planar Hall effect sensors: Evolution, optimization, and applications. *Journal of Physics D: Applied Physics*, 54(35), 353001. <https://doi.org/10.1088/1361-6463/abfbfb>

307. Garrett, J., Luis, E., Peng, H.-H., Cera, T., Gobinathj, Borrow, J., Keçeci, M., Splines, Iyer, S., Liu, Y., cju, & Gasanov, M. (2022–2023). SciencePlots (Versions 2.1.1, 2.1.0, 2.0.1). Zenodo. <https://doi.org/10.5281/zenodo.10206719> (v2.1.1); <https://doi.org/10.5281/zenodo.7986336> (v2.1.0); <https://doi.org/10.5281/zenodo.7394724> (v2.0.1)
308. Berber S., Tomanek D. (2009). Hydrogen-induced disintegration of fullerenes and nanotubes: An ab initio study. *Physical Review B, Condensed Matter*, 80(7), 075427. <https://doi.org/10.1103/PhysRevB.80.075427>
309. Ay, M., Etyemez A. (2020). Optimization of the effects of wire EDM parameters on tolerances. *Emerging Materials Research*. <https://doi.org/10.1680/jemmr.20.00076>
310. Osman Ö., Ozdemir O. K., Ulusoy I. et al. (2010). Effect of Ti sublayer on the ORR catalytic efficiency of dc magnetron sputtered thin Pt films. *International Journal of Hydrogen Energy*, 35(10), 4466-4473. <https://doi.org/10.1016/j.ijhydene.2010.02.077>
311. Keçeci, M. (2025). Echoes of Constancy: Waves of Change in the Keçeci and Oresme Sequences. In *SciELO Preprints*. <https://doi.org/10.1590/SciELOPreprints.12584>
312. Keçeci, M. (2025). Kuantum Hata Düzeltmede Metrik Seçimi ve Algoritmik Optimizasyonun Büyük Ölçekli Yüzey Kodları Üzerindeki Etkileri. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15572200>
313. Keçeci, M. (2025). Kuantum Hata Düzeltme Algoritmalarında Özyineleme Optimizasyonu ve Aşırı Gürültü Toleransı: Kuantum Sıçraması Potansiyelinin Değerlendirilmesi. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15570678>
314. Keçeci, M. (2025). Yüksek Kübit Sayılı Kuantum Hesaplama Ölçeklenebilirlik ve Hata Yönetimi: Yüzey Kodları, Topolojik Malzemeler ve Hibrit Algoritmik Yaklaşımlar. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15558153>
315. Keçeci, M. (2025). Künneth Teoremi Bağlamında Özdevinimli ve Evrişimli Kuantum Algoritmalarında Yapay Zekâ Entegrasyonu ile Hata Minimizasyonu. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15540875>
316. Keçeci, M. (2025). The Relationship Between Gravitational Wave Observations and Quantum Computing Technologies. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15524251>
317. Keçeci, M. (2025). Kütleçekimsel Dalga Gözlemleri ile Kuantum Bilgisayar Teknolojileri Arasındaki Teknolojik ve Metodolojik Bağlantılar. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15519591>
318. Keçeci, M. (2025). Accuracy, Noise, and Scalability in Quantum Computation: Strategies for the NISQ Era and Beyond. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15515113>
319. Keçeci, M. (2025). Quantum Error Correction Codes and Their Impact on Scalable Quantum Computation: Current Approaches and Future Perspectives. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15499657>
320. Keçeci, M. (2025). Nanoscale Quantum Computers Fundamentals, Technologies, and Future Perspectives. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15493024>
321. Keçeci, M. (2025). Investigating Layered Structures Containing Weyl and Majorana Fermions via the Stratum Model. *Open Science Articles (OSAs)*, Zenodo. <https://doi.org/10.5281/zenodo.15489074>

322. Keçeci, M. (2025). Diversity of Keçeci Numbers and Their Application to Prešić-Type Fixed-Point Iterations: A Numerical Exploration. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15481711>
323. Keçeci, M. (2025). Kuantum geometri, topolojik fazlar ve yeni matematiksel yapılar: Disiplinlerarası bir perspektif. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15474957>
324. Keçeci, M. (2025). Understanding quantum mechanics through Hilbert spaces: Applications in quantum computing. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15468754>
325. Keçeci, M. (2025). Nodal-line semimetals: A geometric advantage in quantum information. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15455271>
326. Keçeci, M. (2025). Weyl semimetals: Discovery of exotic electronic states and topological phases. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15447116>
327. Keçeci, M. (2025, May 15). The Keçeci binomial square: A reinterpretation of the standard binomial expansion and its potential applications. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15425529>
328. Keçeci, M. (2025, May 14). Kececisquares. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15411670>
329. Keçeci, M. (2025, May 13). Scalable complexity: Mathematical analysis and potential for physical applications of the Keçeci circle fractal. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15392772>
330. Keçeci, M. (2025, May 13). Kececifractals. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15392518>
331. Keçeci, M. (2025, May 11). Keçeci numbers and the Keçeci prime number: A potential number theoretic exploratory tool. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15381697>
332. Keçeci, M. (2025, May 10). Kececinumbers. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15377659>
333. Keçeci, M. (2025). From Majorana fermions to quantum devices: The role of nanomaterials in the second quantum era. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15331067>
334. Keçeci, M. (2025, May 1). Keçeci Layout. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15314328>
335. Keçeci, M. (2025, May 1). Kececilayout. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.15313946>
336. Keçeci, M. (2025, May 6). Grikod. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.12731345>
337. Keçeci, M. (2011). $2n$ -dimensional at Fujii model instanton-like solutions and coupling constant's role between instantons with higher derivatives. Turkish Journal of Physics, 35(2), 173–178.
<https://doi.org/10.3906/fiz-1012-66>

338. Keçeci, M. (2021). Öz Farkındalık: Mindfulness (Bilgeliğin Üçüncü Adımı: Third Step of Wisdom). ISBN: 9781034850311, Blurb
339. Keçeci, M. (2021). The Next Stop: Future Planet Walks. In SEDS Space Arts 2021, Global Art Competition, Sri Lanka. <https://doi.org/10.13140/RG.2.2.21394.12482>
340. Keçeci, M. (2019). Quantum and Art. Presented at International Workshop on Quantum Frontiers of Technology, TÜBİTAK, TÜSSİDE, Gebze, Türkiye. <https://doi.org/10.13140/RG.2.2.27533.90089>
341. Keçeci, M. (2019, December 6). 2 Boyutlu Tek Katmanlı Yapıların Su Arıtımında Kullanımının Stratejik Önemi [Strategic Importance of Use of 2 Dimensional Monolayer Structures in Water Purification] [Conference presentation]. 23. Sıvı Hâl Sempozyumu (23rd Liquid State Symposium), Piri Reis University, Türkiye. <https://doi.org/10.5281/zenodo.15567811>; <https://www.researchgate.net/publication/337812505>
342. Keçeci, M. (2017, July 19–21). Açık Dijital Rozetlerin Eğitim ve Kariyer Planlamasında Kullanımı [Use of open digital badges in education and career planning] [Conference presentation]. ADIM Fizik Günleri VI, Balıkesir Üniversitesi (ADIM Physics Days VI, Balıkesir University), Türkiye. <https://doi.org/10.5281/zenodo.15567962>; <https://adimfizikvi.balikesir.edu.tr>; <https://www.researchgate.net/publication/318658786>
343. Keçeci, M. (2005, September 13–16). 2n-boyutlu Fujii modelinde instanton çözümleri ve bağlantı sabitinin instantonlar arasındaki rolü. Presented at World Year of Physics 2005 Turkish Physical Society 23rd International Physics Congress, Muğla University, Türkiye. <https://dx.doi.org/10.13140/RG.2.1.1441.4887>
344. Keçeci, M. (2005, May). Konformal invariant Fujii modelinin instanton tipi tam çözümü [Instanton-like exact solution of the conformal invariant Fujii model] [Conference presentation]. Traditional Erzurum Physics Days-II, Atatürk University, Türkiye. <https://dx.doi.org/10.13140/RG.2.1.3538.6408>
345. Keçeci, M. (2002, September 16–20). Exact instanton-like solution conformal invariant of Fujii model, construct for four-dimensional and subderivative [Conference presentation]. Presented at Working Group II, Turkish Nonlinear Science Working Group, Karaburun/Izmir, Türkiye. <https://dx.doi.org/10.13140/RG.2.1.1638.0964>
346. Keçeci, M. (2021). Aşkın Anatomisi: Anatomy of Love, ISBN: 9781034515982, Blurb
347. Keçeci, M. (2025, May 6). Grikod2. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15352206>
348. Keçeci, M. (2025). Kütleçekimsel Dalga Gözlemleri ile Kuantum Bilgisayar Teknolojileri Arasındaki Teknolojik ve Metodolojik Bağlantılar. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15519591>
349. Keçeci, M. (2025). The Relationship Between Gravitational Wave Observations and Quantum Computing Technologies. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15524251>
350. Gottesman, D. (1997). Stabilizer codes and quantum error correction. <https://doi.org/10.48550/arXiv.quant-ph/9705052>
351. Terhal, B. M. (2015). Quantum error correction for quantum memories. Reviews of Modern Physics, 87(2), 307.

352. Dennis, E., Kitaev, A., Landahl, A., & Preskill, J. (2002). Topological quantum memory. *Journal of Mathematical Physics*, 43(9), 4452-4505.
353. Sweke, R., Kesselring, M. S., van Nieuwenburg, E.P.L. Eisert, J. (2020). Reinforcement learning decoders for fault-tolerant quantum computation. *Machine Learning: Science and Technology*, 2, 2. [arXiv:1812.08451v5] <https://doi.org/10.1088/2632-2153/abc609>
354. Nautrup, H.P., Delfosse, N., Dunjko, V., Briegel, H.J., and Friis, N. (2019). Optimizing Quantum Error Correction Codes with Reinforcement Learning. *Quantum* 3, 215. [arXiv:1812.08451v5] <https://doi.org/10.22331/q-2019-12-16-215>
355. Keçeci, M. (2025). Keçeci Numbers and the Keçeci Prime Number. Authorea. <https://doi.org/10.22541/au.174890181.14730464/v1>
356. Keçeci, M. (2025). Bridging Quantum Theory and Computation: The Role of Hilbert Spaces. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.38.1>; <https://doi.org/10.48546/workflowhub.document.38.2>; <https://doi.org/10.48546/workflowhub.document.38.3>
357. Keçeci, M. (2025). Hilbert Spaces and Quantum Information: Tools for Next-Generation Computing. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29604011>
358. Keçeci, M. (2025). Between Chaos and Order: A Behavioural Portrait of Keçeci and Oresme Numbers. preprints.ru. <https://doi.org/10.24108/preprints-3113623>
359. Keçeci, M. (2025). oresmej [Data set]. ResearchGate. <https://doi.org/10.13140/RG.2.2.30518.41284>
360. Keçeci, M. (2025). oresmej [Data set]. figshare. <https://doi.org/10.6084/m9.figshare.29554532>
361. Keçeci, M. (2025). oresmej [Data set]. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/WORKFLOWHUB.DATFILE.19.1>
362. Keçeci, M. (2025). oresmej. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15874178>
363. Keçeci, M. (2025). Analysing the Dynamic and Static Structures of Keçeci and Oresme Sequences. Authorea. <https://doi.org/10.22541/au.175199926.64529709/v1>
364. Keçeci, M. (2025). Dynamic Sequences Versus Static Sequences: Keçeci and Oresme Numbers in Focus. Preprints. <https://doi.org/10.20944/preprints202507.0781.v1>
365. Keçeci, M. (2025). Mobility and Constancy in Mathematical Sequences: A Study on Keçeci and Oresme Numbers. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/osf.io/68r4v>
366. Keçeci, Mehmet (2025). Dynamic and Static Approaches in Mathematics: Investigating Keçeci and Oresme Sequences. Knowledge Commons. <https://doi.org/10.17613/gbdgx-d8y63>
367. Keçeci, Mehmet (2025). Dynamic-Static Properties of Keçeci and Oresme Number Sequences: A Comparative Examination. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29504960>
368. Keçeci, M. (2025). Variability and Stability in Number Sequences: An Analysis of Keçeci and Oresme Numbers. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.37.1>; <https://doi.org/10.48546/workflowhub.document.37.2>

369. Keçeci, M. (2025). Dynamic vs Static Number Sequences: The Case of Keçeci and Oresme Numbers. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15833351>
370. Keçeci, M. (2025). A Graph-Theoretic Perspective on the Keçeci Layout: Structuring Cross-Disciplinary Inquiry. Preprints. <https://doi.org/10.20944/preprints202507.0589.v1>
371. Keçeci, M. (2025). Oresme. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29504708>
372. Keçeci, M. (2025). Oresme [Data set]. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.datafile.18.1>
373. Keçeci, M. (2025). Oresme (0.1.0). Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15833238>
374. Keçeci, M. (2025). Exploring Weyl Semimetals: Emergence of Exotic Electrons and Topological Order. HAL open science. <https://hal.science/hal-05146435>; <https://doi.org/10.13140/RG.2.2.35594.17606>
375. Keçeci, M. (2025). The Rise of Weyl Semimetals: Exotic States and Topological Transitions. Authorea. <https://doi.org/10.22541/au.175192231.19609379/v1>
376. Keçeci, M. (2025). Geometric Resilience in Quantum Systems: The Case of Nodal-Line Semimetals. Authorea. Authorea. <https://doi.org/10.22541/au.175192307.76278430/v1>
377. Keçeci, M. (2025). Harnessing Geometry for Quantum Computation: Lessons from Nodal-Line Materials. Knowledge Commons. <https://doi.org/10.17613/w6vmd-4vb84>
378. Keçeci, M. (2025). Quantum Information at the Edge: Topological Opportunities in Nodal-Line Materials. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29484947>
379. Keçeci, M. (2025). Nodal-Line Semimetals: Unlocking Geometric Potential in Quantum Information. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.36.1>
380. Keçeci, M. (2025). From Weyl Fermions to Topological Matter: The Physics of Weyl Semimetals. Knowledge Commons. <https://doi.org/10.17613/p79v7-kje79>
381. Keçeci, M. (2025). Weyl Semimetals and Their Unique Electronic and Topological Characteristics. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29483816>
382. Keçeci, M. (2025). Weyl Semimetals: Unveiling Novel Electronic Structures and Topological Properties. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.35.3>
383. Keçeci, M. (2025). When Nodes Have an Order: The Keçeci Layout for Structured System Visualization. HAL open science. <https://hal.science/hal-05143155>; <https://doi.org/10.13140/RG.2.2.19098.76484>
384. Keçeci, M. (2025). The Keçeci Layout: A Cross-Disciplinary Graphical Framework for Structural Analysis of Ordered Systems. Authorea. <https://doi.org/10.22541/au.175156702.26421899/v1>
385. Keçeci, M. (2025). Beyond Traditional Diagrams: The Keçeci Layout for Structural Thinking. Knowledge Commons. <https://doi.org/10.17613/v4w94-ak572>
386. Keçeci, M. (2025). The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29468135>

387. Keçeci, M. (2025, July 3). The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/9HTG3>
388. Keçeci, M. (2025). Beyond Topology: Deterministic and Order-Preserving Graph Visualization with the Keçeci Layout. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.34.4>
389. Keçeci, M. (2025). The Keçeci Layout: A Structural Approach for Interdisciplinary Scientific Analysis. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15792684>
390. Keçeci, M. (2025). Technical and Theoretical Bridges Between Gravitational Wave Observations and Quantum Information Processing Systems. Authorea. July, 2025. <https://doi.org/10.22541/au.175138854.46819184/v1>
391. Keçeci, M. (2025). New Technological and Methodological Approaches in Gravitational Wave Detection and Quantum Computing Development. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.33.1>
392. Keçeci, M. (2025). Scalable Complexity in Fractal Geometry: The Keçeci Fractal Approach. Authorea. June, 2025. <https://doi.org/10.22541/au.175131225.56823239/v1>
393. Keçeci, M. (2025). Keçeci Fractals. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.32.2>
394. Keçeci, M. (2025). Keçeci Deterministic Zigzag Layout. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.31.1>
395. Keçeci, M. (2025). Keçeci Zigzag Layout Algorithm. Authorea. <https://doi.org/10.22541/au.175087581.16524538/v1>
396. Keçeci, M. (2025). Keçeci's Arithmetical Square. Authorea. <https://doi.org/10.22541/au.175070836.63624913/v1>
397. Keçeci, M. (2025). Stratum Model-Based Analysis of Topological Insulators Hosting Weyl and Majorana Fermions. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.39.2>; <https://doi.org/10.48546/workflowhub.document.39.1>
398. Keçeci, M. (2025). Quantum Computing Applications of Weyl-Majorana Hybrid States in Layered Systems via Stratum Model. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29606039>
399. Keçeci, M. (2025). Characteristic Features of Keçeci and Oresme Number Sequences: Dynamic and Static Perspectives. HAL open science, hal-05169251. <https://doi.org/10.13140/RG.2.2.24879.85922>
400. Keçeci, M. (2025). Kuantum Algoritmalarında Veri Kodlama ve Kuantizasyon Arasındaki İlişkinin Analizi ve Keçeci Layout ile Max-Cut Problemi. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16755186>
401. Keçeci, M. (2025). Keçeci Varsayımının Kuramsal ve Karşılaştırmalı Analizi. ResearchGate. <https://dx.doi.org/10.13140/RG.2.2.21825.88165>
402. Keçeci, M. (2025). Genelleştirilmiş Keçeci Operatörleri: Collatz Yinelemesinin Nötrosifik ve Hiperreel Sayı Sistemlerinde Uzantıları. Authorea. <https://doi.org/10.22541/au.175433544.41244947/v1>

403. Keçeci, M. (2025). From Abstract Theory to Practical Application: The Journey of Hilbert Space in Quantum Technologies. Preprints. <https://doi.org/10.20944/preprints202508.0171.v2>;
<https://doi.org/10.20944/preprints202508.0171.v1>
404. Keçeci, M. (2025). The Unifying Role of Hilbert Space in Quantum Field Theory and Information Science. Authorea. <https://doi.org/10.22541/au.175449372.28574879/v1>;
<https://doi.org/10.22541/au.175433455.53782703/v1>
405. Keçeci, M. (2025). Keçeci ve Collatz Karşılaştırması: Benzer Algoritmalar, Farklı Çekiciler. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29815910>
406. Keçeci, M. (2025). Keçeci Varsayımı'nın Hesaplanabilirliği: Sonlu Adımda Kararlı Yapıya Yakınsama Sorunu. WorkflowHub. <https://doi.org/10.48546/workflowhub.document.44.1>;
<https://doi.org/10.48546/workflowhub.document.44.2>
407. Keçeci, M. (2025). Keçeci Varsayımı ve Dinamik Sistemler: Farklı Başlangıç Koşullarında Yakınsama ve Döngüler. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/68AFN>
408. Keçeci, M. (2025). Keçeci Varsayımı: Periyodik Çekiciler ve Keçeci Asal Sayısı (KPN) Kavramı. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/g60hy-egx74>
409. Keçeci, M. (2025). Hilbert Space: The Mathematical Engine of Quantum Information Processing. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/6gagh-4dw41>
410. Keçeci, M. (2025). Hilbert Space as the Geometric Foundation of Quantum Mechanics and Computing. OSF. <https://doi.org/10.17605/OSF.IO/ZXDBK>
411. Keçeci, M. (2025). Keçeci Varsayımı: Collatz Genelleştirilmesi Olarak Çoklu Cebirsel Sistemlerde Yinelemeli Dinamikler. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16702475>
412. Keçeci, M. (2025). The Keçeci Layout: A Deterministic Visualisation Framework for the Structural Analysis of Ordered Systems in Chemistry and Environmental Science. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16696713>
413. Keçeci, M. (2025). oresmen. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16634186>
414. Keçeci, M. (2025). The Signature of a Sequence: Variability and Stability in Keçeci and Oresme Numbers. ScienceOpen Preprints. <https://doi.org/10.14293/PR2199.001860.v1>
415. Keçeci, M. (2025). Döngülerden Vektörleştirmeye: Harmonik Seriler için Saf Python ve JAX Performans Karşılaştırması. Authorea. <https://doi.org/10.22541/au.175390609.94042878/v1>
416. Keçeci, M. (2025). From Loops to Vectorisation: A Performance Comparison of Pure Python and JAX for Harmonic Series Calculation. Authorea. <https://doi.org/10.22541/au.175390610.08488249/v1>
417. Keçeci, M. (2025). Keçeci Sayılarının Nötrosifik Çerçeve de Hipergerçek Dönüşümleri ve Uygulamaları. Authorea. <https://doi.org/10.22541/au.175390599.93612305/v1>
418. Keçeci, M. (2025). Hyperreal Transformations and Applications of Keçeci Numbers in a Neutrosophic Framework. Authorea. <https://doi.org/10.22541/au.175390600.02906392/v1>
419. Keçeci, M. (2025). Hipergerçek Analiz ve Nötrosifik Kümelere Dayalı Keçeci Sayılarının Dinamik Modellenmesi. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/jy9mn-2va66>

420. Keçeci, M. (2025). Dynamic Modelling of Keçeci Numbers Based on Hyperreal Analysis and Neutrosophic Sets. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/n4cqw-efp22>
421. Keçeci, M. (2025). Harmonik Seri Hesaplamalarının Modernizasyonu: Geleneksel Python ve JAX Arasında Bir Performans Kıyaslaması. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/BT5A3>
422. Keçeci, M. (2025). Modernising the Computation of Harmonic Series: A Performance Benchmark between JAX and Traditional Python. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/56JDU>
423. Keçeci, M. (2025). Hesaplamalı Matematikte Verimlilik ve Sürdürülebilirlik: Harmonik Seri İçin JAX Tabanlı Bir Yaklaşım. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/bfw58-cbm15>
424. Keçeci, M. (2025). Efficiency and Sustainability in Computational Mathematics: A JAX-Based Approach to the Harmonic Series. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/js67q-4wc71>
425. Keçeci, M. (2025). Hesaplamalı Matematikte Python'un Sınırları ve JAX ile Genişletilmesi: Harmonik Sayılar Üzerine Bir Uygulama. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.42.2>
426. Keçeci, M. (2025). The Limits of Python in Computational Mathematics and Their Extension with JAX: An Application on Harmonic Numbers. WorkflowHub. <https://doi.org/10.48546/workflowhub.document.43.1>
427. Keçeci, M. (2025). Performans ve Ölçeklenebilirlik Analizi: Harmonik Seri Hesaplamalarında JAX ve Saf Python'un Karşılaştırılması. figshare. <https://doi.org/10.6084/m9.figshare.29666675>
428. Keçeci, M. (2025). A Comparative Analysis of Performance and Scalability: Computing Harmonic Series with JAX versus Pure Python. figshare. <https://doi.org/10.6084/m9.figshare.29666684>
429. Keçeci, M. (2025). A Comparative Study of Pure Python and JAX-Based Approaches in Computing Harmonic Series. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16576092>
430. Keçeci, M. (2025). Harmonik Serilerin Hesaplanmasında Saf Python ve JAX Tabanlı Yaklaşımların Karşılaştırılması. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16536195>
431. Keçeci, M. (2025). The Keçeci Layout: A Deterministic, Order-Preserving Visualization Algorithm for Structured Systems. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16526798>
432. Keçeci, M. (2025). Keçeci Sayılarının Nötrosifik ve Hipergerçek Uzaylarda Geometrik Analizi. WorkflowHub. <https://doi.org/10.48546/workflowhub.document.40.1>
433. Keçeci, M. (2025). Geometric Interpretations of Keçeci Numbers within Neutrosophic and Hyperreal Number Systems. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.41.1>
434. Keçeci, M. (2025). Keçeci Sayılarının Nötrosifik Hipergerçek Uzaylarda Geometrik Temsilleri. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29636750>

435. Keçeci, M. (2025). Geometric Representations of Keçeci Numbers in Neutrosophic Hyperreal Spaces. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29636849>
436. Keçeci, M. (2025). Keçeci Sayılarının Nötrosifik Küme Teorisi ve Hipergerçek Uzaylarda İncelenmesi. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/KVCB6>
437. Keçeci, M. (2025). Investigation of Keçeci Numbers via Neutrosophic Set Theory and Hyperreal Spaces. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/OSF.IO/VMK82>
438. Keçeci, M. (2025). Geometric Interpretations of Keçeci Numbers with Neutrosophic and Hyperreal Numbers. Zenodo. <https://doi.org/10.5281/zenodo.16344232>
439. Keçeci, M. (2025). Keçeci Sayılarının Nötrosifik ve Hipergerçek Sayılarla Geometrik Yorumlamaları. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16343568>
440. Keçeci, M. (2025). adnus [Data set]. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/osf.io/9c26y>
441. Keçeci, M. (2025). adnus [Data set]. Open Fig Share Articles (OFSAs). figshare. <https://doi.org/10.6084/m9.figshare.29621336>
442. Keçeci, M. (2025). adnus [Data set]. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.datafile.23.1>
443. Keçeci, M. (2025). adnus. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16341919>
444. Keçeci, M. (2025). Characterization of Keçeci Number Systems as Chaotic and Hyperchaotic Maps. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16954468>
445. Keçeci, M. (2025). Deterministic Visualization of Distribution Power Grids: Integration of Power Grid Model and Keçeci Layout. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16934620>
446. Keçeci, M. (2025). Interactive Exploration of the Hamiltonian Problem with Z3 and the Keçeci Layout. Open Fig Share Articles (OFSAs), figshare. <https://doi.org/10.6084/m9.figshare.29959778>
447. Keçeci, M. (2025). An Interactive Tool for Graph Theory Education: Exploring the Hamiltonian Problem with Z3 and the Keçeci Layout. Open Science Output Articles (OSOAs), OSF. <https://doi.org/10.17605/osf.io/hzu8y>
448. Keçeci, M. (2025). The Hamiltonian Problem in Graph Theory Education: An Interactive Approach Using Z3 and the Keçeci Layout. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/mvq42-h4262>
449. Keçeci, M. (2025). Solving the Hamiltonian Problem in Graph Theory Education with Z3 and the Keçeci Layout. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.48.2>
450. Keçeci, M. (2025). Hamiltonian Problem with Z3 and the Keçeci Layout. ResearchGate. <https://doi.org/10.13140/RG.2.2.27327.78244>
451. Keçeci, M. (2025). A Novel Tool for Graph Theory Education: Interactive Exploration of the Hamiltonian Problem with Z3 and the Keçeci Layout. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.16920991>
452. Keçeci, M. (2025). Z3 ve Keçeci Layout ile Hamilton Problemi. ResearchGate. <https://doi.org/10.13140/RG.2.2.23316.97924>

453. Keçeci, M. (2025). Graf Teorisi Eğitiminde Yeni Bir Araç: Z3 ve Keçeci Yerleşimi ile Hamilton Probleminin İnteraktif Keşfi. Open Fig Share Articles (OFSAs), figshare.
<https://doi.org/10.6084/m9.figshare.29958116>
454. Keçeci, M. (2025). Graf Teorisi Eğitiminde Yeni Bir Araç: Z3 ve Keçeci Layout ile Hamilton Probleminin İnteraktif Keşfi. Open Science Output Articles (OSOAs), OSF.
<https://doi.org/10.17605/osf.io/e23us>
455. Keçeci, M. (2025). Graf Teorisi Eğitiminde Z3 ve Keçeci Layout ile Hamilton Problemi. Open Science Knowledge Articles (OSKAs), Knowledge Commons. <https://doi.org/10.17613/g5r9k-ksb90>
456. Keçeci, M. (2025). Graf Teorisi Eğitiminde Z3 ve Keçeci Dizilimi ile Hamilton Problemi. Open Work Flow Articles (OWFAs), WorkflowHub. <https://doi.org/10.48546/workflowhub.document.45.2>
457. Keçeci, M. (2025). Graf Teorisi Eğitiminde Yeni Bir Araç: Z3 ve Keçeci Dizilimi ile Hamilton Probleminin İnteraktif Keşfi. Open Science Articles (OSAs), Zenodo.
<https://doi.org/10.5281/zenodo.16883657>
458. Keçeci, M. (2025). Hilbert Space Theory and Its Implementation in Quantum Computing Systems. preprints.ru. <https://doi.org/10.24108/preprints-3113653>
459. Keçeci, M. (2020). Stratum Model [Unpublished pre-doctoral I. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
460. Keçeci, M. (2021). Nano Quantum Computer (nQC) [Unpublished pre-doctoral II. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
461. Keçeci, M. (2021). Quantum Error Correction (QEC) Codes [Unpublished pre-doctoral III. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
462. Keçeci, M. (2022). Accuracy, Noise, and Scalability in Quantum Computation [Unpublished pre-doctoral IV. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
463. Keçeci, M. (2022). Künneth Teoremi Bağlamında Özdevinimli ve Evrişimli Kuantum Algoritmalarında Yapay Zekâ Entegrasyonu ile Hata Minimizasyonu [Unpublished pre-doctoral V. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
464. Keçeci, M. (2023). Yüksek Kübit Sayılı Kuantum Hesaplama Ölçeklenebilirlik ve Hata Yönetimi [Unpublished pre-doctoral VI. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
465. Keçeci, M. (2023). Kuantum Hata Düzeltme Algoritmalarında Özyineleme Optimizasyonu ve Aşırı Gürültü Toleransı [Unpublished pre-doctoral VII. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
466. Keçeci, M. (2024). Kuantum Hata Düzeltmede Metrik Seçimi [Unpublished pre-doctoral VIII. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
467. Keçeci, M. (2024). Çoklu İşlemci Mimarilerinde Kuantum Algoritma Simülasyonlarının Hızlandırılması. Unpublished pre-doctoral IX. report. Gebze Technical University, Kocaeli, Türkiye.
468. Keçeci, M. (2025). Error Minimisation in Autonomous and Convolutional Quantum Algorithms through Artificial Intelligence Integration in the Context of the Künneth Theorem. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.17214806>
469. Keçeci, M. (2022). Error Minimisation in Autonomous and Convolutional Quantum Algorithms through Artificial Intelligence Integration in the Context of the Künneth Theorem [Unpublished pre-doctoral V. technical reports]. Gebze Technical University, Kocaeli, Türkiye.

470. Keçeci, M. (2025). Scalability and Error Management in High-Qubit-Count Quantum Computing: Surface Codes, Topological Materials, and Hybrid Algorithmic Approaches. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.17227501>
471. Keçeci, M. (2022). Scalability and Error Management in High-Qubit-Count Quantum Computing [Unpublished pre-doctoral VI. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
472. Keçeci, M. (2023). Recursion Optimisation and Extreme Noise Tolerance in Quantum Error Correction Algorithms [Unpublished pre-doctoral VII. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
473. Keçeci, M. (2025). Recursion Optimisation and Extreme Noise Tolerance in Quantum Error Correction Algorithms: Assessing the Potential for a Quantum Leap. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.17243336>
474. Keçeci, M. (2024). Metric Selection in Quantum Error Correction [Unpublished pre-doctoral VIII. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
475. Keçeci, M. (2025). The Impact of Metric Selection and Algorithmic Optimisation on Large-Scale Surface Codes in Quantum Error Correction. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.17259861>
476. Keçeci, M. (2021). Nano Kuantum Bilgisayarı (nQC) [Unpublished pre-doctoral II. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
477. Keçeci, M. (2025). Nano Ölçekli Kuantum Bilgisayarlar: Temelleri, Teknolojileri ve Gelecek Perspektifleri. Open Science Articles (OSAs), Zenodo.
478. Keçeci, M. (2025). Çoklu İşlemci Mimarilerinde Kuantum Algoritma Simülasyonlarının Hızlandırılması. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.15580503>
479. Keçeci, M. (2023). Accelerating Quantum Algorithm Simulations in Multi-Processor Architectures [Unpublished pre-doctoral IX. technical reports]. Gebze Technical University, Kocaeli, Türkiye.
480. Keçeci, M. (2025). Accelerating Quantum Algorithm Simulations in Multi-Processor Architectures: Optimisation Techniques with Cython, Numba, and Jax. Open Science Articles (OSAs), Zenodo. <https://doi.org/10.5281/zenodo.17287508>