Struttura della tesi

Introduzione

Contesto generale

- **La simulazione nella HEP**: necessità computazionali crescenti
- **Bottleneck di GEANT4**: lentezza dei metodi Monte Carlo tradizionali
- **Machine Learning come soluzione**: GAN per accelerare la simulazione del calorimetro
- **Problema ingegneristico**: ottimizzazione computazionale ed energetica

Motivazioni della tesi

- **Efficienza computazionale**: BoloGAN e GEANT4 in termini di performance
- **Consumo energetico**: analisi del costo energetico di training e inferenza
- **Sostenibilità**: impatto ambientale delle simulazioni intensive
- **Configurazioni ottimali**: parametri di rete e risorse computazionali

Obiettivi

- Caratterizzazione delle performance computazionali di BoloGAN
- Studio del consumo energetico al variare di configurazioni e risorse
- Confronto quantitativo con GEANT4 (dove possibile)
- Identificazione di configurazioni ottimali (performance vs consumo)

Metodologia seguita

- **Setup sperimentale**: cluster OPH, ambienti containerizzati
- **Metriche di valutazione**: tempo di esecuzione, utilizzo risorse, consumo energetico
- **Design degli esperimenti**: variazione sistematica di parametri

Struttura della tesi

Capitolo 1 – Dalla fisica fondamentale alla simulazione computazionale

1.1. Fisica delle particelle e modello standard

- **Modello Standard**: panoramica delle particelle fondamentali
- **Interazioni fondamentali**: elettromagnetica, debole, forte
- **Rivelabilità indiretta**: come "vedere" particelle invisibili attraverso i rivelatori
- **Obiettivo della ricerca**: scoprire l'invisibile attraverso l'osservabile

1.2. Il CERN e l'esperimento ATLAS

- **CERN**: missione e ruolo nella ricerca fondamentale
- **LHC**: struttura dell'acceleratore, energie in gioco
- **Esperimenti principali**: CMS, ALICE, LHCb, ATLAS
- **ATLAS**: struttura, sottosistemi, produzione di dati (ordini di grandezza)

1.3. Il calorimetro di ATLAS e la necessità della simulazione

- **Calorimetria**: principi fisici di funzionamento

- **Struttura del calorimetro**: elettromagnetico e adronico
- **Segmentazione spaziale**: coordinate η, φ e voxelizzazione
- **Output dati**: rappresentazione 3D degli impatti, volumi di dati
- **Perché simulare**: validazione teorica, ottimizzazione del rivelatore

1.4. GEANT4: simulazione fisica di dettaglio

- **Metodi Monte Carlo**: simulazione stocastica, campionamento statistico
- **Architettura GEANT4**: moduli, fisica implementata
- **Precisione vs complessità**: simulazione step-by-step delle interazioni
- **Costo computazionale**: tempi di calcolo, requisiti di memoria
- **Dati quantitativi**: ore CPU per evento, proiezioni future

1.5. L'infrastruttura computazionale del CERN

1.5.1. Worldwide LHC Computing Grid (WLCG)

- **Modello distribuito**: necessità del calcolo distribuito per LHC
- **Architettura Tier**: Tier-0 (CERN), Tier-1, Tier-2, Tier-3
- **Workflow computazionale**: job submission, data distribution, analysis
- **Scale del problema**: PetaBytes di dati, milioni di job/anno

1.5.2. Problematiche computazionali crescenti

- **HL-LHC impact**: aumento di luminosità e volume dati (10x)
- **Consumo energetico**: stime attuali e proiezioni future
- **Sostenibilità**: necessità di soluzioni più efficienti
- **Bottleneck identificato**: necessità di simulazioni più rapide

1.6. Simulazione veloce con AtlFast3

- **Motivazioni**: necessità di simulazione veloce per analisi di fisica
- **Architettura ibrida**: FastCaloSim V2 + FastCaloGAN
- **Trade-off**: velocità vs precisione
- **Risultati**: speed-up factors rispetto a GEANT4

1.7. BoloGAN e il CaloChallenge

- **Motivazioni INFN Bologna**: miglioramenti rispetto a FastCaloGAN
- **CaloChallenge**: framework di validazione, dataset standard
- **Obiettivi**: maggiore efficienza e facilità di deployment
- **Contesto della tesi**: focus su ottimizzazione computazionale ed energetica

---

Capitolo 2 – Reti neurali e sistemi computazionali

2.1. Fondamenti delle reti neurali

- **Il neurone artificiale**: modello matematico, funzione di attivazione
- **Architetture**: feedforward, layer, connessioni
- **Apprendimento**: backpropagation, ottimizzazione del gradiente
- **Teorema dell'approssimazione universale**: capacità espressiva
- **Reti profonde**: motivazioni, vantaggi, limitazioni

2.2. Le GAN: teoria e applicazione

2.2.1. Motivazioni e principi

- **Limiti dei modelli generativi classici**: difficoltà nella generazione di dati complessi

- **Nascita delle GAN**: Goodfellow et al. (2014)
- **Min-max game**: generatore vs discriminatore
- **Equilibrio di Nash**: convergenza teorica

2.2.2. Architettura e training

- **Generatore**: trasformazione rumore → dato realistico
- **Discriminatore**: classificazione reale vs artificiale
- **Training alternato**: algoritmo di addestramento
- **Funzioni di loss**: GAN standard, miglioramenti successivi

2.2.3. Miglioramenti delle GAN

- **Wasserstein GAN**: distanza di Wasserstein, stabilità
- **Gradient Penalty**: regolarizzazione del discriminatore
- **Conditional GAN**: controllo della generazione tramite etichette
- **Applicazioni in fisica**: motivazioni specifiche per la calorimetria

2.3. BoloGAN: architettura e implementazione

- **Struttura della rete**: generatore e discriminatore
- **Dataset e preprocessing**: voxelizzazione, normalizzazione
- **Configurazione**: parametri, binning, modularità
- **Compatibilità ATLAS**: integrazione con pipeline esistenti

2.4. Infrastrutture software e gestione distribuita

2.4.1. Problematiche del deployment HPC

- **Eterogeneità**: diversità hardware/software tra cluster
- **Dipendenze**: complessità delle librerie scientifiche
- **Riproducibilità**: necessità di ambienti consistenti
- **Scalabilità**: gestione risorse multi-nodo

2.4.2. Stack tecnologico

- **Conda**: gestione pacchetti e ambienti virtuali
- **TensorFlow**: framework di deep learning, ottimizzazioni GPU
- **Apptainer**: containerizzazione per HPC, integrazione CUDA
- **SLURM**: resource manager, job scheduling, backfilling
- **ROOT**: framework di analisi dati per fisica delle particelle

2.4.3. Integrazione e ottimizzazione

- **Container workflow**: build, deployment, execution
- **GPU acceleration**: CUDA runtime, driver compatibility
- **Distributed training**: multi-GPU, multi-nodo
- **Performance tuning**: batch size, memory management, I/O optimization

---

Capitolo 3 – Analisi computazionale ed energetica

3.1. Setup sperimentale e metodologia

3.1.1. Ambiente di test

- **Cluster OPH**: descrizione hardware (CPU, GPU, memoria, rete)
- **Configurazione software**: container Apptainer, stack TensorFlow/CUDA
- **Ambiente controllato**: isolamento job, riproducibilità

3.1.2. Metodologia di misurazione

- **Metriche computazionali**: tempo di esecuzione, throughput, utilizzo risorse
- **Metriche energetiche**: potenza istantanea, energia totale, efficienza
- **Strumenti di monitoring**: nvidia-smi, htop, power meters/RAPL
- **Protocollo sperimentale**: warm-up, ripetizioni, significatività statistica

3.2. Ottimizzazione del data pipeline

3.2.1. Formato dati e I/O

- **Migrazione CSV → Parquet**: motivazioni, implementazione
- **Analisi prestazionale**: tempi di caricamento, compressione
- **Ottimizzazione lettura**: engine PyArrow, parallelizzazione
- **Impatto su training**: riduzione overhead I/O

3.2.2. Preprocessing e caching

- **Pipeline di preprocessing**: normalizzazione, augmentation
- **Strategie di caching**: dataset in memoria, intermediate results
- **Batch processing**: dimensioni ottimali, prefetching

3.3. Esperimenti computazionali

3.3.1. Parametri di configurazione testati

- **Architettura rete**: numero layer, dimensioni, funzioni attivazione
- **Training parameters**: batch size, learning rate, optimizer
- **Resource allocation**: CPU threads, GPU memory, distributed settings

3.3.2. Test single-node

- **CPU vs GPU**: confronto prestazioni, analisi bottleneck
- **Scaling con batch size**: throughput, memory usage, stabilità
- **Ottimizzazioni TensorFlow**: XLA compilation, mixed precision
- **Profiling dettagliato**: hotspot identification, resource utilization

3.3.3. Test multi-node (se applicabile)

- **Distributed training**: data parallelism, communication overhead
- **Scaling efficiency**: strong scaling, weak scaling
- **Network impact**: bandwidth usage, latency effects

3.4. Analisi energetica

3.4.1. Metodologia di misurazione energetica

- **Hardware instrumentation**:
  - **RAPL (Running Average Power Limit)**: contatori CPU integrati
  - **nvidia-smi**: monitoraggio potenza GPU
  - **Power meters esterni**: misure di sistema completo (se disponibili)
- **Granularità temporale**: campionamento continuo vs discreto
- **Baseline establishment**: idle power, peak power

3.4.2. Metriche energetiche

- **Potenza media**: durante training, durante inferenza
- **Energia per epoca**: integrale potenza × tempo
- **Efficienza energetica**: performance/watt, samples/Joule
- **Energy-delay product**: trade-off tempo vs energia

3.4.3. Fattori di influenza

- **CPU frequency scaling**: analisi DVFS, performance vs power

- **GPU boost clocks**: comportamento dinamico, thermal throttling
- **Memory subsystem**: impatto accessi memoria su consumo
- **Workload characteristics**: compute-intensive vs memory-bound

3.5. Risultati sperimentali

3.5.1. Performance analysis

- **Training time**: tempo per epoca, convergenza
- **Throughput**: samples/second, events/hour
- **Resource utilization**: CPU/GPU occupancy, memory usage
- **Scalabilità**: comportamento al variare delle risorse

3.5.2. Energy consumption analysis

- **Power profiles**: andamento potenza nel tempo
- **Energy breakdown**: CPU vs GPU vs memory
- **Efficiency metrics**: confronto diverse configurazioni
- **Optimal operating points**: configurazioni Pareto-ottimali

3.5.3. Quality assessment

- **Accuracy metrics**: χ^2/NDF, distribution matching
- **Physics validation**: confronto con GEANT4 (dati di riferimento)
- **Trade-off analysis**: accuratezza vs performance vs energia

3.6. Confronto con GEANT4

3.6.1. Baseline GEANT4

- **Dati di riferimento**: consumi energetici da letteratura
- **Normalizzazione**: events/hour, energy/event
- **Limitazioni**: disponibilità dati, configurazioni diverse

3.6.2. Comparative analysis

- **Speed-up factor**: BoloGAN vs GEANT4
- **Energy efficiency**: riduzione consumo energetico
- **Total Cost of Ownership**: proiezioni costi operativi
- **Scalabilità**: comportamento con HL-LHC volumes

3.7. Discussione e ottimizzazioni

3.7.1. Identificazione bottleneck

- **Computational bottlenecks**: CPU vs GPU vs I/O bound
- **Energy hotspots**: componenti più energivori
- **Optimization opportunities**: margini di miglioramento

3.7.2. Configurazioni ottimali

- **Pareto frontiers**: performance vs energy vs accuracy
- **Production recommendations**: configurazioni consigliate
- **Sensitivity analysis**: robustezza rispetto ai parametri

3.8. Prospettive future e ottimizzazioni avanzate

3.8.1. Ottimizzazioni immediate

- **Model compression**: pruning, quantization
- **Hardware acceleration**: specializzazione GPU, TPU
- **Algorithm improvements**: training strategies, architectures

3.8.2. Considerazioni architetturali

- **ARM vs x86**: panoramica vantaggi energetici ARM
- **Edge computing**: distribuzione calcolo, latency vs energy
- **Neuromorphic computing**: prospettive ultra-low power

3.8.3. Integration roadmap

- **ATLAS workflow**: integrazione operativa
- **Monitoring infrastructure**: deployment production
- **Sustainability goals**: target riduzione impatto ambientale

---

Conclusioni

Riepilogo del percorso sperimentale

- **Obiettivi raggiunti**: caratterizzazione computazionale ed energetica di BoloGAN
- **Metodologia applicata**: approccio sistematico, misure quantitative
- **Risultati principali**: performance improvements, energy efficiency gains

Valutazione dell'efficienza computazionale e sostenibile

- **Performance comparison**: BoloGAN vs GEANT4 speed-up factors
- **Energy analysis**: riduzione consumo energetico, efficiency metrics
- **Quality retention**: mantenimento accuratezza fisica
- **Scalability assessment**: comportamento con risorse crescenti

Contributo fornito

- **Caratterizzazione quantitativa**: prima analisi sistematica energy/performance
- **Optimization insights**: identificazione configurazioni ottimali
- **Deployment guidelines**: raccomandazioni per produzione
- **Methodology**: framework riutilizzabile per analisi simili

Prospettive future

Sviluppi immediati

- **Model optimizations**: pruning, quantization, mixed precision
- **Hardware acceleration**: specializzazione GPU, utilizzo TPU
- **Production deployment**: integrazione workflow ATLAS

Prospettive tecnologiche

- **Architetture ARM**: potenziale efficienza energetica superiore
- **Edge computing**: distribuzione calcolo, riduzione latenza
- **Green computing**: integrazione energie rinnovabili, carbon neutrality

Impatto a lungo termine

- **Sostenibilità HEP**: contributo alla riduzione footprint energetico
- **Scalabilità HL-LHC**: preparazione per volumi dati futuri
- **Methodology transfer**: applicabilità ad altri domini scientifici

---

Studio di fattibilità: ottimizzazioni avanzate

ALTA PRIORITÀ (Fattibilità alta, Impatto alto)

1. Quantizzazione mixed precision

- Conversione pesi da float32 a float16/int8 usando tf.keras.mixed_precision
- Metodi: automatic mixed precision, manual casting, performance comparison
- Difficoltà: Bassa, supporto nativo TensorFlow
- Effort: 1-2 settimane, risultati immediati su energia e velocità
- Metriche: speed-up factor, memory reduction, energy saving, accuracy retention

2. XLA (Accelerated Linear Algebra) compilation

- Ottimizzazione automatica del grafo computazionale TensorFlow
- Metodi: jit_compile=True, confronto performance con/senza XLA
- Difficoltà: Bassa, flag di configurazione
- Effort: 1 settimana, setup immediato
- Metriche: compilation overhead, runtime speed-up, energy efficiency

3. Inferenza on-demand vs batch processing

- Confronto strategie di inferenza per scenari produzione realistici
- Metodi: single event latency vs batch throughput, different batch sizes
- Difficoltà: Media, richiede setup di test diversificati
- Effort: 2 settimane, analisi comprehensive
- Metriche: latency distribution, throughput, energy-per-event, GPU utilization

MEDIA PRIORITÀ (Fattibilità media, Impatto medio-alto)

4. GPU memory optimization

- Ottimizzazione utilizzo memoria GPU per efficiency
- Metodi: memory growth, optimal batch sizing, memory profiling
- Difficoltà: Media, richiede tuning empirico
- Effort: 1-2 settimane, testing iterativo
- Metriche: memory utilization, OOM prevention, performance scaling

5. Container overhead analysis

- Misurazione impatto containerizzazione su performance native
- Metodi: native vs container benchmarks, startup time, resource isolation
- Difficoltà: Media, richiede setup dual environment
- Effort: 1 settimana, comparison framework
- Metriche: performance overhead, startup latency, resource efficiency

6. Advanced data pipeline optimization

- Ottimizzazione tf.data pipeline per riduzione I/O bottleneck
- Metodi: prefetching, parallel loading, memory mapping, caching strategies
- Difficoltà: Media, richiede profiling dettagliato
- Effort: 2 settimane, iterative optimization
- Metriche: loading time, pipeline efficiency, CPU utilization

BASSA PRIORITÀ (Fattibilità variabile, Impatto futuro)

7. Checkpointing intelligente

- Ottimizzazione frequenza salvataggio stato per fault tolerance
- Metodi: checkpoint overhead analysis, optimal frequency determination
- Difficoltà: Alta, implementazione custom complex
- Effort: 3+ settimane, engineering intensivo
- Metriche: checkpoint overhead, recovery time, storage requirements

8. Distributed training energy analysis

- Estensione analisi energetica a training multi-nodo
- Metodi: multi-GPU scaling, communication overhead measurement
- Difficoltà: Alta, richiede resource allocation significative
- Effort: 2-3 settimane, dipende da availability cluster
- Metriche: scaling efficiency, communication energy cost, total energy scaling

PROSPETTIVE FUTURE (Out of scope tesi, research directions)

9. Model compression avanzata

- Pruning strutturato, knowledge distillation, neural architecture search
- Applicabilità: Troppo research-oriented per tesi triennale
- Timeline: 6+ mesi development cycle

10. Hardware acceleration specializzato

- FPGA implementation, neuromorphic computing, quantum-classical hybrid
- Applicabilità: Richiede hardware non disponibile
- Timeline: Multi-year research projects

11. Edge computing deployment

- Distributed inference, federated learning, mobile deployment
- Applicabilità: Fuori scope ATLAS workflow
- Timeline: Architectural redesign required

Validation metrics avanzate

Metriche statistiche robuste

- **Wasserstein-1 distance**: earth mover distance tra distribuzioni reali e generate
- **Kolmogorov-Smirnov test**: test non-parametrico per uguaglianza distribuzioni
- **Energy-per-1000-events**: metrica normalizzata per confronto diretto con GEANT4
- **Performance-per-watt**: throughput computazionale per unità di potenza