# El Agente Gráfico: Structured Execution Graph for Scientific Agents

**Jiaru Bai** [1,2] * **Abdulrahman Aldossary** [1,2] * **Thomas Swanick** [3,4] **Marcel Müller** [1,2]
**Yeonghun Kang** [1,2] **Zijian Zhang** [1,3] **Jin Won Lee** [5] **Tsz Wai Ko** [1,2] **Varinia Bernales** [1,3,6]
**Alán Aspuru-Guzik** [1,2,3,6,7,8,9,10]

*Equal contribution [1]*Department of Chemistry, University of Toronto, 80 St. George St., Toronto, ON M5S 3H6, Canada* [2]*Vector Institute for Artificial Intelligence, W1140-108 College St., Schwartz Reisman Innovation Campus, Toronto, ON M5G 0C6, Canada* [3]*Department of Computer Science, University of Toronto, 40 St George St., Toronto, ON M5S 2E4, Canada* [4]*Department of Mathematics, University of Toronto, 40 St George St., Toronto, ON M5S 2E4, Canada* [5]*School of Computer Science, McGill University, 3480 University St., Montréal, QC H3A 0E9, Canada* [6]*Acceleration Consortium, 700 University Ave., Toronto, ON M7A 2S4, Canada* [7]*Department of Materials Science & Engineering, University of Toronto, 184 College St., Toronto, ON M5S 3E4, Canada* [8]*Department of Chemical Engineering & Applied Chemistry, University of Toronto, 200 College St., Toronto, ON M5S 3E5, Canada* [9]*Canadian Institute for Advanced Research (CIFAR), 661 University Ave., Toronto, ON M5G 1M1, Canada* [10]*NVIDIA, 431 King St. W #6th, Toronto, ON M5V 1K4, Canada*. Correspondence to: Varinia Bernales varinia@bernales.org, Alán Aspuru-Guzik alan@aspuru.com.

## 1. Introduction

Recent advances in large language models (LLMs) have enabled agents to reliably invoke external tools and parameterise functional calls [1, 2, 3, 4]. Many systems adopt multi-agent architectures to manage context across heterogeneous tools. However, this design is fundamentally misaligned with scientific workflows, where the central challenge is the reliable propagation of large, structured numerical state across computational pipelines, particularly in GPU-accelerated simulations [5]. Encoding such data within LLM context windows is impractical and exacerbates coordination and verification overheads intrinsic to multi-agent execution [6].

We introduce El Agente Gráfico, a single-agent framework that embeds LLM planning within a **type-safe execution graph**. Instead of transmitting execution state through free-form text, scientific state is represented as typed Python objects managed by an object graph mapper (OGM) and persisted to a knowledge graph (KG). This enables symbolic references, explicit provenance, caching, and validated tool orchestration while keeping the LLM context strictly control-oriented.

## 2. Gráfico architecture

Figure 1 illustrates the El Agente Gráfico architecture. The core abstraction is a *typed execution graph* exposed to the agent as a tool, enabling structural context reduction for GPU-accelerated scientific workflows. Rather than encoding the intermediate state as free-form text, Gráfico represents computation as a directed graph of state-transformation nodes with statically validated inputs and outputs.

Using GPU4PySCF as an example, its mean-field objects encapsulate substantial internal state that is computationally expensive to recompute and impractical to transmit through the LLM's context window. Gráfico addresses this by re-casting the computational capability abstractions of El Agente Q [1] into graph nodes whose edges encode admissible data flow, allowing execution state to persist independently of LLM reasoning.

Gráfico adopts `ConceptualAtoms` as a unified data structure across heterogeneous tools, including geometry generation (e.g., RDKit), multiplicity verification (QCElemental), and electronic structure calculations (GPU4PySCF). It enables zero-copy transfer of molecular state, such as passing CREST conformer ensembles directly to downstream PySCF workflows.

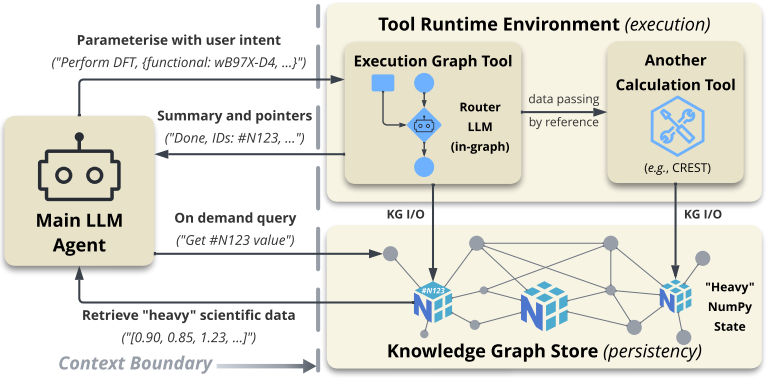


Fig. 1: Overall architecture of El Agente Gráfico, a single-agent system with a typed execution graph and persistent KG. User interaction triggers LLM-controlled execution over a graph of admissible computation nodes, where typed state transformations operate on shared in-memory objects. Routing decisions are made via constrained LLM calls, while general-purpose tools support KG access, code execution, and web search.

Table 1: Performance benchmarking of LLMs within the GRÁFICO architecture. Final context window saturation, retry cost, and carryover tokens respectively measure end-of-task context pressure, overhead from error-driven retries, and the persistence of historical context across agentic trace, jointly characterising an agent's efficiency in managing long-horizon executions.

| LLM model | Numerical eval. (%) | LLM Judge eval. (%) | Trace tokens | LLM requests | Token cost (USD) | Task duration (s) | Final context window saturation (%) | Retry cost (%) | Carryover tokens (%) |
|---|---|---|---|---|---|---|---|---|---|
| EL AGENTE Q [1] | *88.25 with human eval* | | 1,649,616 | 168.43 | 4.67 | 1,827 | n/a | n/a | n/a |
| gpt-4.1 | 93.71 | 96.52 | 113,175 | 5.11 | 0.25 | 208 | **3.59** | 8.17 | 66.79 |
| gpt-5 | **98.88** | **98.50** | **83,613** | **3.32** | 0.17 | 228 | 9.78 | 9.59 | **53.21** |
| gpt-5.1 | 98.37 | 96.61 | 99,775 | 3.65 | 0.17 | 211 | 10.43 | 8.69 | 58.19 |
| gpt-5.2 | 98.69 | 96.19 | 95,153 | 4.40 | 0.22 | **180** | 8.42 | 8.27 | 64.59 |
| minimax-m2 | 89.66 | 87.61 | 210,018 | 6.40 | **0.05** | 315 | 21.93 | 16.38 | 78.60 |
| qwen3-max | 93.54 | 90.33 | 178,671 | 3.84 | 0.24 | 296 | 26.27 | 11.82 | 61.45 |
| sonnet-3.7 | 93.69 | 90.94 | 284,036 | 9.14 | 0.92 | 404 | 19.46 | **5.07** | 86.30 |
| sonnet-4.5 | 96.07 | 95.67 | 320,397 | 6.58 | 1.09 | 273 | 26.79 | 9.02 | 83.28 |

Execution begins when the agent invokes the graph tool with a summarised user intent and calculation parameters. When multiple successor nodes are admissible, routing is delegated to a lightweight controller implemented as a constrained LLM call. By exposing the schemas of valid successor nodes as the controller's output schema, routing is framed as a structured output-generation problem, ensuring both control flow and arguments are schema-compliant by construction. Intermediate results are tracked via the OGM, allowing large numerical objects to propagate across stages without repeated serialisation.

From a memory perspective within the CoALA framework [7], GRÁFICO unifies procedural, semantic, and episodic representations within the execution graph and its associated object store. The execution graph encodes procedural knowledge, while the OGM provides persistent semantic and episodic state. General-purpose tools for web search, sandboxed code execution, and KG interaction complement domain-specific workflows, enabling both short-horizon reasoning and long-term state retrieval.

The system is implemented using `pydantic-ai`, leveraging static type hints and runtime validation to enforce explicit interfaces between control logic and execution. All inter-node communication is schema-driven, allowing the underlying LLM to be swapped dynamically without affecting execution correctness. A reflective error-recovery layer captures failures and augments them with domain-specific diagnostics, enabling informed retries and strategy revisions.

## 3. Benchmark

Compared to EL AGENTE Q [1], which adopts a multi-agent architecture driven by `sonnet-3.7`, GRÁFICO achieves substantially higher efficiency in both cost and time. Operational cost drops from \$4.67 to as low as \$0.17 with `gpt-5` ($\approx$ 96% reduction), while wall-clock time decreases from $1,827$ s to $200 - 300$ s ($\geq$ 6x speedup), alongside a reduction in trace token consumption from $\sim$ 1.6M to 100k, as summarised in Table 1. These gains arise primarily from the single-agent execution-graph design, which eliminates inter-agent communication overhead that would otherwise amplify LLM API usage in multi-agent systems. The reduction in execution time further demonstrates GRÁFICO's ability to exploit parallelised, GPU-accelerated PySCF workflows without incurring coordination latency. For computational chemistry workflows characterised by sequential reasoning and strict numerical validation, these results imply that once a single agent reaches a high baseline accuracy, additional multi-agent coordination is unlikely to yield proportional performance gains. This aligns with findings in [8], where a single token-efficient agent outperforms multi-agent settings in sequential reasoning tasks.

When evaluated under an identical execution graph, accuracy differences across stronger LLMs narrow, while substantial disparities in token usage, retry cost, and context carryover persist, indicating that long-horizon performance depends more on interaction style and state management than on raw reasoning capability. We observe systematic differences between the GPT and Claude model families: GPT models tend to batch and parallelise tool invocations before reasoning over aggregated results, whereas Claude models more frequently interleave reasoning with sequential tool calls. Within the Claude family, `sonnet-3.7` shows limited parallelism, while `sonnet-4.5` incurs high overhead from repeated use of general-purpose code execution. These patterns suggest that system-level efficiency depends on how well a model's interaction style aligns with tool abstractions. Consequently, encapsulating recurrent post-processing into specialised tools can yield further gains independently of model scaling.

## 4. Conclusions

Across benchmarks on university-level quantum chemistry tasks previously run with a multi-agent system, GRÁFICO reliably plans and executes parallel workflows with substantially reduced token usage. We further apply it to Boltzmann-weighted spectroscopic properties in solution and interactive metal-organic framework design, where KGs serve as persistent memory and reasoning substrates. Together, these results suggest that abstraction and type safety offer a more scalable foundation for scientific agents than prompt-centric multi-agent decomposition.

**Acknowledgments**

The authors thank Yiqun Bian for discussions on data analysis and Pit Steinbach for assistance in setting up CREST-GPU. J.B. acknowledges funding from the Eric and Wendy Schmidt AI in Science Postdoctoral Fellowship Program, a program by Schmidt Futures. A.A. acknowledges support from King Abdullah University of Science and Technology (KAUST), Kingdom of Saudi Arabia, for the KAUST Ibn Rushd Postdoctoral Fellowship. T.W.K. acknowledges the support of the Vector Distinguished Postdoctoral Fellowship. A.A.-G. thanks Anders G. Frøseth for his generous support. A.A.-G. also acknowledges the generous support of Natural Resources Canada and the Canada 150 Research Chairs program. This work was supported by the AI2050 program of Schmidt Sciences. This research is part of the University of Toronto's Acceleration Consortium, which receives funding from the CFREF-2022-00042 Canada First Research Excellence Fund. This research was enabled in part by support provided by SciNet HPC Consortium (`https://scinethpc.ca/`) and the Digital Research Alliance of Canada (`https://www.alliancecan.ca`). Computations were performed on the Trillium supercomputer at the SciNet HPC Consortium. SciNet is funded by: the Canada Foundation for Innovation; the Government of Ontario; Ontario Research Fund - Research Excellence; and the University of Toronto.

**References**

[1] Yunheng Zou, Austin H. Cheng, Abdulrahman Aldossary, Jiaru Bai, Shi Xuan Leong, Jorge Arturo Campos-Gonzalez-Angulo, Changhyeok Choi, Cher Tian Ser, Gary Tom, Andrew Wang, Zijian Zhang, Ilya Yakavets, Han Hao, Chris Crebolder, Varinia Bernales, and Alán Aspuru-Guzik. El Agente: An autonomous agent for quantum chemistry. *Matter*, 8(7):102263, July 2025.

[2] Keyan Ding, Jing Yu, Junjie Huang, Yuchen Yang, Qiang Zhang, and Huajun Chen. SciToolAgent: A knowledge-graph-driven scientific agent for multitool integration. *Nature Computational Science*, 5(10):962–972, 2025.

[3] Yeonghun Kang and Jihan Kim. ChatMOF: An artificial intelligence system for predicting and generating metal-organic frameworks using large language models. *Nature Communications*, 15(1), June 2024.

[4] Shuxiang Cao, Zijian Zhang, Mohammed Alghadeer, Simone D Fasciati, Michele Piscitelli, Mustafa Bakr, Peter Leek, and Alán Aspuru-Guzik. Automating quantum computing laboratory experiments with an agent-based AI framework. *Patterns*, 6(10), 2025.

[5] Xiaojie Wu, Qiming Sun, Zhichen Pu, Tianze Zheng, Wenzhi Ma, Wen Yan, Yu Xia, Zhengxiao Wu, Mian Huo, Xiang Li, Weiluo Ren, Sheng Gong, Yumin Zhang, and Weihao Gao. Enhancing GPU-acceleration in the Python-based simulations of chemistry frameworks. *WIREs Computational Molecular Science*, 15(2), 2025.

[6] Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? *arXiv preprint*, (arXiv:2503.13657), 2025.

[7] Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. Cognitive architectures for language agents. *arXiv preprint*, (arXiv:2309.02427), 2023.

[8] Yubin Kim, Ken Gu, Chanwoo Park, Chunjong Park, Samuel Schmidgall, A. Ali Heydari, Yao Yan, Zhihan Zhang, Yuchen Zhuang, Mark Malhotra, Paul Pu Liang, Hae Won Park, Yuzhe Yang, Xuhai Xu, Yilun Du, Shwetak Patel, Tim Althoff, Daniel McDuff, and Xin Liu. Towards a science of scaling agent systems. *arXiv preprint*, (arXiv:2512.08296), 2025.