Luca Nicoli, *PhD*

SELECTED CASE STUDIES · CRYPTOECONOMICS · MEV · RESEARCH ENGINEERING

luca.nicoli.engineer@gmail.com / Pisa, Italy / LinkedIn GitHub Linktree

PROFILE

CURRENT
Research Scientist · *CryptoEconLab*
Cryptoeconomic mechanism design, tokenomics, incentive systems

PREVIOUS
MEV Scientist Engineer · *Urani.trade*
Intent-based solvers, MEV mitigation on Solana

PHD · CUM LAUDE
Theoretical & Computational Chemistry · *Scuola Normale Superiore*
Pisa, Italy

MSC & BSC
Engineering Physics · *Politecnico di Milano*
Milan, Italy

10 JOURNAL ARTICLES

1 INTERNATIONAL PATENT

5 CONFERENCE TALKS

6 CASE STUDIES

INDEX OF CASE STUDIES

01

TOKENOMICS

REVNET

The client. Revnet is a protocol developed by the Juicebox team for launching autonomous, tokenized revenue networks: deterministic rules set at deployment govern how tokens are issued, how value accrues, and how participants access liquidity, with no governance and no intermediaries.

The problem. With every rule immutable at deployment, founders and investors needed to know precisely how these systems behave across parameter configurations before committing capital.

What we did. We combined formal mathematical modeling with computational simulation across parameter configurations, and proved four guarantees: a cash-out price floor that rises mechanically with network activity, issuance schedules that act as tunable monetary policy, guaranteed redemption under all conditions, and loans that remain solvent even under complete default. We also showed that seven immutable parameters determine all system dynamics, making launch-time tuning the single most critical design decision.

DELIVERABLES

- **Public analysis:** Revnets: A Cryptoeconomic Analysis (CryptoEconLab Insights).
- **Companion paper I:** *Cryptoeconomics of Revnets*, the formal mathematical framework.
- **Companion paper II:** *Revnet Parameters Analysis*, computational experiments and design guidance for founders and investors.

02
PROTOCOL ECONOMICS

HELLAS

The client. Hellas is a fully decentralized network for verifiable off-chain tensor compute: it connects clients who need AI computation with providers who supply it, replacing trust with economic guarantees. Its core innovation, the Catgrad compiler, makes execution deterministic across environments, so any incorrect computation can be pinpointed and proven on-chain through succinct fraud proofs.

The problem. Providers post collateral as a commitment to correct execution. For an open compute market to work, cheating must carry negative expected value: under which stake, slashing, and dispute conditions is honest execution the equilibrium, and when do clients find it worthwhile to challenge incorrect results?

What we did. We specified the protocol's economic structure: actors, value flows over the state-channel lifecycle, and fee distribution. We then analyzed the fraud game, deriving the minimum viable provider stake that eliminates the impunity region and characterizing detection as an inspection game with a mixed-strategy equilibrium. Finally, we validated the theory with an agent-based simulation over heterogeneous strategic populations, stress-testing reputation farming, Sybil attacks, collusion, griefing, and censorship.

DELIVERABLES

- **Hellas Economic Analysis:** actors, value flows, and incentive conditions.
- **Fraud Game Analysis:** enforcement and detection equilibria, with design recommendations on stake floors, reward routing, permissionless challenging, randomized audits, and timeouts.
- **ABM Simulation Report:** experimental validation and adversarial analysis, with quantitative parameterization recommendations.
- **Public write-up:** Hellas: Trustless Tensor Compute Through Economic Guarantees.

03
INCENTIVE MECHANISM

The client. Filecoin is the largest decentralized storage network. Filecoin Plus (FIL+) is its incentive layer: DataCap granted to verified clients multiplies the storage power and rewards of providers who store useful data. This work was funded through Filecoin ProPGF.

The problem. Accessing FIL+ DataCap has traditionally been permission-first: applications, off-chain verification, and manual approval before any DataCap is granted.

What we did. We designed and developed AutoCap, an action-first allocation mechanism. Participants earn DataCap by settling Filecoin Pay rails on-chain; finalized FIL burns are irreversible, publicly verifiable proof of real economic activity, and each round distributes a fixed DataCap pool proportionally to measured burns. A small registration fee prevents spam, and a public real-time dashboard makes every allocation independently verifiable on the block explorer.

DELIVERABLES

- **Deployed mechanism and smart contracts:** github.com/CELtd/filplus-autocap.
- **Public dashboard and docs:** filautocap.xyz; launch write-up on CryptoEconLab Insights.
- **Rollout:** validated end-to-end on Calibration testnet; staged mainnet launch via the Experimental Pathway Metallocator, with rounds scaling 10 → 50 → 100 TiB of DataCap.

04
RESEARCH TOOLING

The problem. Reasoning about the Filecoin economy, from storage-provider ROI to supply dynamics and what-if scenarios, required spreadsheet analysis or custom modeling, putting forecasts out of reach for most users. This work was funded through Filecoin ProPGF.

What we did. We designed and developed EconoLens, an MCP-based conversational tool. An LLM layer (DeepSeek V3) parses plain-English questions, an orchestration layer invokes the right tool, and MechaFIL, our differentiable digital twin of the Filecoin economy, computes the forecast. Users receive a plain-language explanation plus interactive charts with CSV export.

DELIVERABLES

- **Live tool:** cryptoeconlab.com/econolens (public pilot).
- **Open-source modeling engine:** github.com/celtd/mechafil-jax.
- **Launch write-up:** EconoLens: Conversational Filecoin Economic Analysis.

Aleph, MEV Solver Agent · *Urani.trade*

MEV MITIGATION & DEFI · INTENT-BASED SOLVERS · SOLANA · PYTHON · APACHE-2.0

OPEN SOURCE

05

MEV INFRA

The context. Urani.trade is an intent-based trading protocol on Solana: users express swap intents that are batched and settled by competing solver agents, an architecture that reduces toxic MEV at the application layer, protecting users from sandwiching and improving execution. I worked on this as MEV Scientist Engineer on the Agents team.

The problem. The protocol needed an in-house solver: an agent able to ingest order batches, match compatible intents peer-to-peer, and route the remainder across fragmented AMM liquidity, fast enough to compete for order flow.

What we did. I co-authored Aleph, Urani's in-house arbitrage agent. Aleph fetches order batches from the orderbook, extracts the intents, settles compatible orders through peer-to-peer matching, spins a thread per unmatched intent to compute best quotes via AMM arbitrage (Jupiter routing in this release), and packs the solutions back to the protocol.

DELIVERABLES

- **Open-source agent template:** github.com/urani-trade/solana-mev-agent-py (Apache-2.0).
- **Modular codebase:** agents, Solana wrappers, order/intent processing, liquidity venues, P2P matching, price oracles (Helius, Pyth, Dexscreener).
- **Tooling:** deployment CLI, local protocol-mimicking server for end-to-end testing, pytest suite, Poetry packaging.

06
OPTIMAL ROUTING

The problem. Given a user's order intent (buy token B, sell token A, at no worse than a limit rate), how should a solver distribute execution across a market of many liquidity venues to maximize the user's surplus?

What we did. The agent models the market as a graph: tokens are vertices, venues with their liquidity pools are edges. From the intent it constructs a strategy, the directed subgraph of paths connecting the sold token to the bought one, and optimizes the flow through each path. Because the forward-routing problem is convex, sequential least-squares programming (scipy's SLSQP) provably finds the optimum, with no need for a global optimizer. Partial fills and strict fly-or-kill orders are both supported.

DELIVERABLES

- **Research codebase:** github.com/luca-nik/mev_agent.
- **Reference abstractions:** Order, Venue, Market, and Agent components mapping one-to-one onto protocol concepts, a readable reference implementation for solver design.