

Uniswap vs CoW Swap

Two different approaches to trade execution

Author
ribo_

Prepared for
Republic Legion Research Forum

Date
30 Sep 2026

The useful comparison is no longer simply "AMM vs solver."

Uniswap combines routing across v2/v3/v4 pools with an additional auction-based execution path through UniswapX. CoW Protocol makes signed intents, solver competition and batch auctions central to its execution design.

Research question

How do Uniswap and CoW Swap differ in the way they source liquidity, select execution, manage MEV exposure, and present execution complexity to the trader?

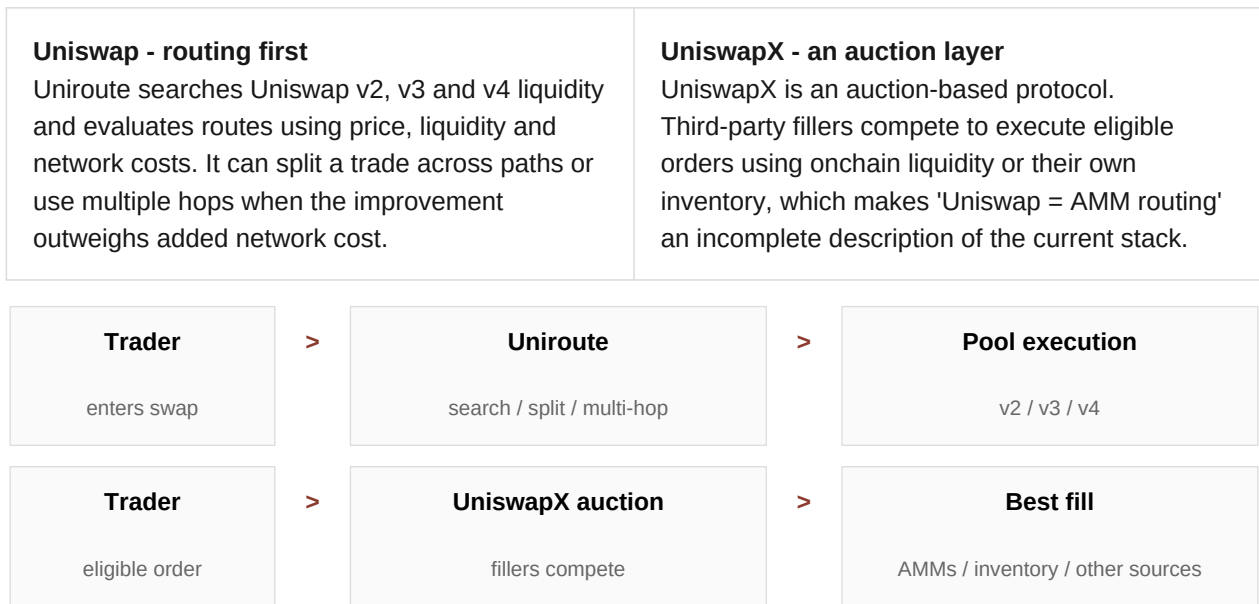
Comparison at a glance

Dimension	Uniswap	CoW Swap / CoW Protocol
Execution	Uniroute searches v2/v3/v4 liquidity; eligible trades can also use UniswapX auctions.	Users sign orders; solvers compete within batch-auction based execution.
Liquidity	Direct, multi-hop and split routes across Uniswap pools; UniswapX can aggregate additional sources through fillers.	Coincidence of Wants can match compatible intents; solvers can also tap onchain liquidity.
MEV	UniswapX provides front-running protection and filler-based execution; direct routing and UniswapX are distinct paths.	Batch auctions and delegated execution reduce simple public-mempool ordering games.
UX / gas	Classic swaps resemble direct onchain execution; UniswapX can move most network costs to fillers.	Users submit signed orders while solvers handle settlement, shifting execution complexity away from the trader.
Core emphasis	Find an efficient execution path across available venues.	Create competition over how a signed intent is settled.

Working conclusion: Uniswap is best described as a routing-first architecture with an intent/auction layer, while CoW is an intent-and-batch-auction architecture from the start.

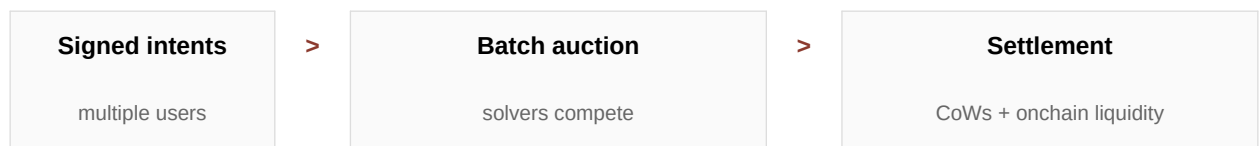
1. EXECUTION ARCHITECTURE

Different starting points, increasing overlap



CoW Protocol - intent and solver competition as the core

CoW Protocol uses fair combinatorial batch auctions as its price-finding mechanism. Users submit signed orders, and solvers search for a settlement that can combine compatible intents and external onchain liquidity. Coincidence of Wants is important because compatible user intents can offset one another before external liquidity is used for any remaining imbalance.



Key distinction: both systems can use competition and abstraction, but CoW makes the multi-order auction central to the protocol, while Uniswap's product stack combines pool routing with UniswapX as an additional execution mode.

What changes for the trader?

Liquidity sourcing Uniswap: Uniroute can split and multi-hop across v2/v3/v4 pools, searching for better effective output after costs. CoW: solvers can first exploit Coincidence of Wants and then use available onchain liquidity when needed.	Price discovery Uniswap: route optimization is performed for the individual swap, with a selected path shown to the user. CoW: batch auctions let solver competition operate across a set of orders, allowing matching and settlement decisions to be considered together.
MEV protection UniswapX advertises front-running protection as a benefit of filler-based auction execution. CoW's batch-auction and delegated-execution design similarly aims to protect traders from simple transaction-ordering games.	Gas and execution UX A conventional Uniswap pool swap still resembles direct onchain execution. In UniswapX, fillers pay most network costs on the swapper's behalf. CoW shifts final settlement work to solvers after the user signs an order.

Why the old one-line comparison breaks down

"Uniswap routes; CoW uses solvers" is useful as a first approximation, but it is too binary for the current products. UniswapX introduces an auction and filler competition, which overlaps conceptually with some of the abstraction commonly associated with solver-based systems.

More precise framing:

Uniswap = routing-first architecture + optional intent/auction execution.

CoW = intent-first architecture + batch-auction settlement.

What this means for research

- Compare the actual execution mode used for a specific trade, not just the protocol brand.
- Separate where liquidity comes from from who decides how the trade is executed.
- Treat MEV protection and gas abstraction as execution-path properties, not universal labels.
- For UX analysis, ask whether the interface makes these distinctions understandable without overwhelming the user.

3. UX IMPROVEMENT PROPOSAL

Execution Summary: transparency without overload

The research exposed a practical UX problem: two swaps can look nearly identical in a confirmation screen even when the execution model behind them is materially different. A lightweight, optional Execution Summary could make this visible without turning the swap interface into a technical dashboard.

Suggested default view

Expected output	1,000 USDC
Estimated network cost	\$2.15
Execution label	Intent Auction

Expandable Execution details

Liquidity source	Multiple pools / filler inventory / matched intents
MEV protection	Private relay or auction protection used for this trade
Network cost payer	User / filler / solver / included in settlement
Route explanation	One sentence explaining why this execution path was selected

Design principle: progressive disclosure. Keep the default confirmation view simple, then let users who care about execution quality expand the technical details. The label should describe the actual execution chosen for that trade, rather than permanently branding a protocol as one execution model.

Primary sources

Uniswap Labs - How does routing work? (updated Aug 21, 2026)

<https://support.uniswap.org/hc/en-us/articles/46932289118733-How-does-routing-work>

Uniswap Labs - What is UniswapX? (updated Jun 9, 2026)

<https://support.uniswap.org/hc/en-us/articles/17515415311501-What-is-UniswapX>

Uniswap Labs - Who fills UniswapX orders?

<https://support.uniswap.org/hc/en-us/articles/17081681967757-Who-fills-UniswapX-orders>

Uniswap Labs - Are there network costs for UniswapX?

<https://support.uniswap.org/hc/en-us/articles/17544708791821-Are-there-network-costs-for-UniswapX>

CoW Protocol Documentation - Welcome / What is CoW Protocol?

<https://docs.cow.fi/>

Bottom line: Both products aim to improve execution quality, but the path matters. A more transparent confirmation experience would let advanced users understand that path without forcing every user to learn the underlying infrastructure.