

# A Code to Describe Satellite Constellations

---

[draft-piraux-space-constellation-code-01](#)

**Juan A. Fraire** — Inria Lyon / Saarland University

Maxime Piraux — Aerospacelab

IETF 125 · SPACE Research Group · Shenzhen, China · March 2026

## The Challenge

---

Network researchers increasingly study **Space Networks** — but:

- Satellite constellations are **complex** orbital constructs
  - Orbital mechanics, perturbations, coverage geometry, ISL topology
- **Orbital parameters** are publicly tracked (NORAD/TLEs) — but only for existing satellites
  - Future and hypothetical constellations have no TLEs
  - TLEs are position snapshots and do not encode constellation design intent
- **Link topology** is largely **proprietary** (Starlink, OneWeb, Telesat...)
  - ISL structure, capacity, and policies are not publicly disclosed

**Result:** poor reproducibility, heterogeneous assumptions, and high entry barriers.

## Our Goal: A Common Notation

---

Propose a **compact, human-readable code** to describe satellite constellation patterns, specifically targeting the **research community**.

Three design principles:

1. **Simplicity** — encode a full constellation in a single string; no orbital mechanics expertise required
2. **Realism** — based on Walker parameters used by real operators (Iridium, GPS, Starlink, OneWeb...)
3. **Usability** — easily documentable in papers and directly consumable by simulation/analysis tools

**A code should be expressible in a paper caption, an email, or a terminal command.**

# Walker Constellations: Background

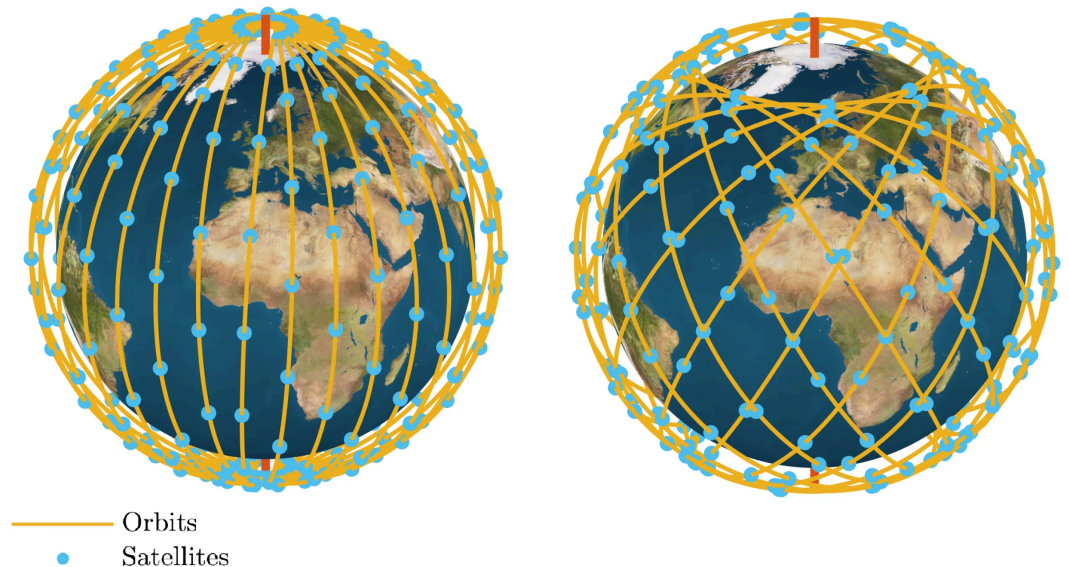
Most LEO/MEO mega-constellations follow a **Walker pattern**: circular orbits sharing the same altitude and inclination.

Variant	RAAN span	Typical inclination	Seam?
Walker Star (S)	180°	~90° (near-polar)	Yes — counter-rotating boundary
Walker Delta (D)	360°	45–65° (mid-latitude)	No — all planes co-rotate

## Key parameters per shell:

- Altitude · Inclination · Total satellites  $T$  · Orbital planes  $P$  · Phasing factor  $F$

The **phasing factor  $F$**  staggers satellites across adjacent planes — controls the coverage fill pattern.



Left: **Walker Star (S)** — Right: **Walker Delta (D)**

Source: Chan et al., *Remote Sensing* 14(17):4232, 2022

# The Constellation Code

A complete orbital shell encoded in **one string**:

```
W : altitude_km : inclination_deg : T / P / F [: M0]
```

Field	Meaning	Constraints
W	D (Delta) or S (Star)	—
altitude_km	Altitude above Earth's surface (km)	> 0
inclination_deg	Orbital inclination (°)	[0, 180]
T	Total number of satellites	T mod P = 0
P	Number of orbital planes	—
F	Phasing factor	[0, P-1]
M <sub>0</sub>	Mean anomaly of first satellite (optional)	[0, 360), default 0

Multi-shell constellations concatenate shells with **+**.

The syntax is formally specified as an **ABNF grammar** (RFC 5234) in the I-D.

## Real-World Examples

---

Constellation	Code	T	P	Walker
GPS	D:20180:55:24/6/1	24	6	Delta — MEO, global GNSS
Iridium	S:780:86.4:66/6/1	66	6	Star — LEO, near-polar
OneWeb	S:1200:87.9:672/12/11	672	12	Star — LEO, near-polar
Starlink (sh. 1)	D:550:53:1584/72/39	1584	72	Delta — LEO, mid-lat

*In some cases the phasing factor is speculative (not publicly available).*

A two-shell example: D:550:53:24/6/0+S:780:86.4:22/11/1

## Describing Inter-Satellite Links (ISL)

---

The code is extended with a **YAML format** to specify link patterns within shells.

```
version: draft-piraux-space-constellation-code-01
shells:
- code: D:1200:55:400/20/19
  link_patterns:
  - rank_offset: 1           # link sat[p][r] → sat[p][r+1]: next sat in same plane
  - plane_offset: 1         # link sat[p][r] → sat[p+1][r']: next plane (staggered)
    conditions:             # limits to 3 links/sat (checkerboard pattern)
    - eq:                   # condition: rank%2 == plane%2
      - {mod: [rank, 2]}    # current satellite's rank index mod 2
      - {mod: [plane, 2]}   # current satellite's plane index mod 2
                           # meaning: only link sat[p][r] → sat[p+1][r'] when rank%2 == plane%2

- code: S:1210:89:52/4/1
  link_patterns:
  - rank_offset: 1          # link sat[p][r] → sat[p][r+1]: in-plane ring only
                           # (no plane_offset: Star seam has counter-rotating planes)
```

- `rank_offset` / `plane_offset` establish **bidirectional** links
- `conditions` let researchers express **degree-constrained topologies** declaratively (e.g. optical terminals limited to 2–4 links per satellite)

## Document Status & Roadmap

---

**Current:** `draft-piraux-space-constellation-code-01`

- Defines Walker Star and Walker Delta shells
- ABNF grammar for the code string and YAML extension for ISL link patterns
- Examples: GPS, Iridium, OneWeb, Starlink

**In scope for future versions:**

- Elliptical orbits (Flower constellations) — currently out of scope
- Optical Communication Terminal (OCT) capabilities — complement topology intent
- Additional predicates and operations in condition expressions

**Seeking feedback from the SPACE RG on:**

- **Completeness** — are there relevant patterns not expressible today?
- **Adoption** — can tools in the community use this format? (i.e., Nishanth's list)
- **Scope** — should the I-D also cover ground segment or inter-shell links?

# Summary

---

**Problem:** researchers lack a common, simple notation for satellite constellations — hurting reproducibility and accessibility.

**Proposal:** a constellation code striking a balance between simplicity and realism:

Layer	What it defines
Code string	Walker type · altitude · inclination · T/P/F
YAML format	Multi-shell · ISL link patterns · conditions
Tooling	Parse · validate · propagate · visualize

One string. One file. One shared language.

**Draft:** `draft-piraux-space-constellation-code` (IRTF SPACE RG)

**Code:** `github.com/mpiraux/draft-piraux-space-constellation-code`

**Tools:** `https://gitlab.inria.fr/jfraire/constellation-code-tools`

# Thank you

---

## Questions & Feedback Welcome

Juan A. Fraire — `juan.fraire@inria.fr`

Maxime Piraux — `maxime.piraux@aerospacelab.com`

*draft-piraux-space-constellation-code · IETF 125 · Shenzhen · March 2026*

## ISL Conditions: Expression Reference

Conditions are an **expression tree** evaluated per satellite filtering which form a cross-plane link.

Variable	Meaning
rank	Satellite index within its orbital plane
plane	Orbital plane index

**Operators** nest arbitrarily — the top-level expression must resolve to a boolean:

Operator	Syntax	Returns
mod	{mod: [value, divisor]}	value % divisor (integer)
eq	{eq: [left, right]}	left == right (bool)

All entries in `conditions:` are AND-ed — every one must be true for the link to be created.

**Example — checkerboard pattern** (limits cross-plane links to alternate satellites):

```
conditions:  
- eq:  
  - {mod: [rank, 2]} # rank % 2  
  - {mod: [plane, 2]} # plane % 2
```

Reads as: create the link only when `rank % 2 == plane % 2`. Combined with one in-plane `rank_offset: 1` link, this caps each satellite at **3 links** total.

## Tooling: `code_run`

---

A reference implementation ships alongside the I-D:

```
Code string / YAML
├── parser + validator → ConstellationSpec (or clear error)
├── geometry.py       → Keplerian elements per satellite
├── links.py          → ISL graph (YAML only)
├── --backend orb     pure Python, J2 propagation, CSV + plots
└── --backend stk     STK 12, J4/HPOP, full 3-D visualization
```

```
# Validate a code string (instant, no computation)
python -m code_run "D:550:53:1584/72/39" --backend orb --validate-only

# Propagate Iridium and plot ground tracks + ISL snapshot
python -m code_run tutorial/examples/02_iridium.yaml --backend orb --plot
```

Outputs: `orbital_elements.csv`, `ground_tracks.png`, `orbits_3d.png`, `ground_tracks_isl.png`