contactgraphgen

contactgraphgen reads a plain-text *command file* (one command per line) and produces contact data for REPLAY:

- A single header line listing all nodes: NODE_LIST <id_1> <id_2> . . .
- Followed by one line per *edge occurrence*: u, v, <ISO8601StartZ>, <ISO8601EndZ>;

CLI

```
python contactgraphgen.py <commands.txt> --out <graph.txt>
```

Command File Syntax

Each non-empty line contains exactly one command. Lines starting with # are comments and are ignored. Whitespace between tokens is flexible (one or more spaces or tabs).

Commands

ADDNODE

Syntax
ADDNODE <ID>

Adds the integer node ID to the set of known nodes. Re-adding an existing ID has no effect.

ADDEDGE

Syntax
ADDEDGE [ID1,ID2,ID3,...] <UNIX_STARTTIME> <DURATION> <RECURRENCE_DELTA> [OCCURRENCES N UNTIL <ISO8601|UNIX>]

IDs A bracketed, comma-separated list. Optional spaces are allowed: [1,2,3] and [1, 2, 3] are both valid.

Semantics Let the listed IDs form a set S. For each occurrence time

$$t_k = \texttt{UNIX\_STARTTIME} + k \times \texttt{RECURRENCE\_DELTA},$$

the tool creates an edge occurrence between *every unordered pair* $\{u, v\}$ with $u \neq v$ and $u, v \in S$, i.e., a complete graph on S for each occurrence. Each occurrence lasts DURATION seconds, so the end time printed is $t_k + \texttt{DURATION}$.

Per-edge horizon Exactly one of the following may be appended to the ADDEDGE line:

- OCCURRENCES N $\Rightarrow$ generate $k = 0, 1, \ldots, N-1$ (if N=0, nothing is emitted).
- UNTIL T $\Rightarrow$ generate all k such that $t_k < T$. T may be ISO 8601 UTC (e.g., 2006-08-20T15:05:00Z) or a UNIX second.

If neither OCCURRENCES nor UNTIL is present, a single occurrence ($k = 0$) is generated. If RECURRENCE_DELTA is 0, only $k = 0$ is generated regardless of horizon.

Example

Example command file **example.txt**

```
# Define five nodes
ADDNODE 1
ADDNODE 2
ADDNODE 3
ADDNODE 4
ADDNODE 5

# Create a complete-triad contact that starts at UNIX=0, lasts 60s,
# recurs every 120s, with exactly 2 occurrences (k = 0, 1)
ADDEDGE [1,2,3] 0 60 120 OCCURRENCES 2

# Create a pair contact between 4 and 5 that starts at UNIX=60, lasts 45s,
# recurs every 30s until 1970-01-01T00:03:00Z (start times t_k < until)
ADDEDGE [4,5] 60 45 30 UNTIL 1970-01-01T00:03:00Z
```

Run

```
python contactgraphgen.py example.txt --out graph.txt
```

Resulting contact graph **graph.txt**

```
NODE_LIST 1 2 3 4 5
1, 2, 1970-01-01T00:00:00Z, 1970-01-01T00:01:00Z;
1, 3, 1970-01-01T00:00:00Z, 1970-01-01T00:01:00Z;
2, 3, 1970-01-01T00:00:00Z, 1970-01-01T00:01:00Z;
4, 5, 1970-01-01T00:01:00Z, 1970-01-01T00:01:45Z;
4, 5, 1970-01-01T00:01:30Z, 1970-01-01T00:02:15Z;
1, 2, 1970-01-01T00:02:00Z, 1970-01-01T00:03:00Z;
1, 3, 1970-01-01T00:02:00Z, 1970-01-01T00:03:00Z;
2, 3, 1970-01-01T00:02:00Z, 1970-01-01T00:03:00Z;
4, 5, 1970-01-01T00:02:00Z, 1970-01-01T00:02:45Z;
4, 5, 1970-01-01T00:02:30Z, 1970-01-01T00:03:15Z;
```

Notes

- The header lists nodes in ascending order with single spaces.
- Each edge line ends with `;` (semicolon and a trailing space).
- Pairs are ordered with `u < v`. The whole file is sorted by start time; ties break by pair order.
- For `UNTIL`, starts are generated while `t_k < UNTIL`; the end time may pass the horizon.