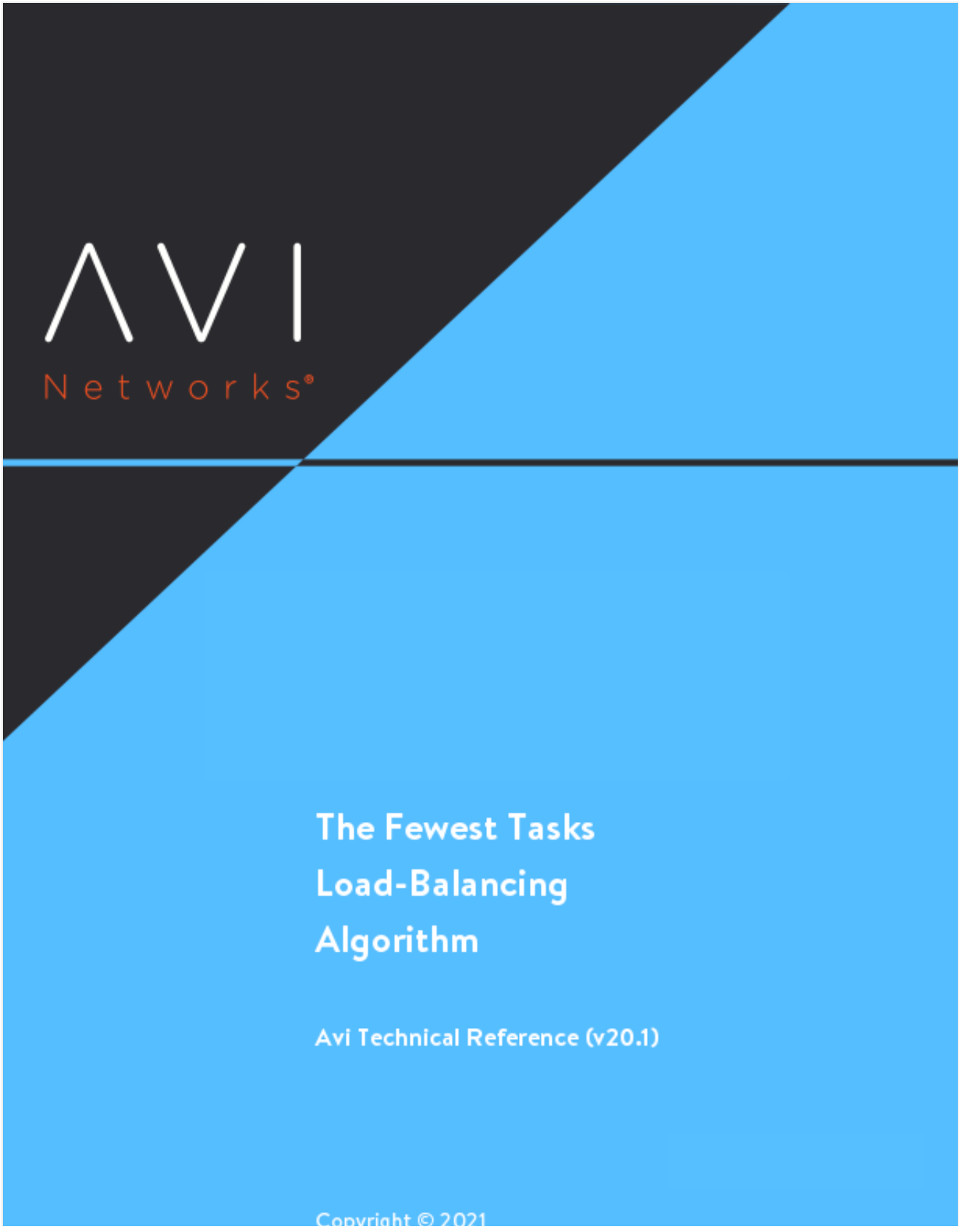
AVI
Networks®
The Fewest Tasks
Load-Balancing
Algorithm
Avi Technical Reference (v20.1)
Copyright © 2021

# The Fewest Tasks Load-Balancing Algorithm

view online

Avi Vantage supports adaptive load balancing based on server feedback, facilitated by an external health monitor. It can be enabled by assigning the `lb_algorithm_fewest_tasks` algorithm to the desired pool. Use either the Avi CLI or Avi REST API; the feature is not visible in the Avi UI.

## CLI Configuration Example

{%cli%} configure pool foo lb_algorithm lb_algorithm_fewest_tasks save {%endcli%}

## How It Works

An external health monitor can feed back a number (for example, 1-100) to the algorithm by writing data into the <hm_name>.<pool_name>.<ip>.<port>.tasks file. Each output from this file would be used to feed back to the algorithm. The range of numbers provided as feedback, and the send interval of the health monitor may be adjusted to tune the load balancing algorithm behavior to the specific environment.

For example, consider a pool `p1` with 2 back-end servers, `s1` and `s2`. Suppose the health monitor ticks every 10 seconds (send-interval), and sends back a feedback of 100 (high load) and 10 (low load). At time `t1`, `s1` and `s2` are set with 100 tasks and 10 tasks respectively. Now, if you send 200 requests, the first 90 would go to s2, since it had "90" more units available. The next 110 would be sent equally to s1 and s2. At time t2 = t1 + 10 sec, `s1` and `s2` get replenished to the new data provided by the external health monitor.

Here is an example script for use by the external health monitor:

```
#!/usr/bin/python
import sys
import httplib
import os
conn = httplib.HTTPConnection(sys.argv[1]+':'+sys.argv[2])
conn.request("GET", "/")
r1 = conn.getresponse()
print r1
if r1.status == 200:
print r1.status, r1.reason ## Any output on the screen indicates SUCCESS for health monitor

try:
fname = sys.argv[0] + '.' + os.environ['POOL'] + '.' + sys.argv[1] + '.' + sys.argv[2] + '.tasks'
f = open(fname, "w")
try:
f.write('230') # Write a string to a file - instead of 230 - find the data from the curl output and feed it.
finally:
f.close()
except IOError:
pass
```

**You can use the** `show pool <foo> detail` **and** `show poo <foo> server detail` **commands to see detailed information about the number of connections being sent to the servers in the pool.**