dm-service-time
===============

dm-service-time is a path selector module for device-mapper targets,
which selects a path with the shortest estimated service time for
the incoming I/O.

The service time for each path is estimated by dividing the total size
of in-flight I/Os on a path with the performance value of the path.
The performance value is a relative throughput value among all paths
in a path-group, and it can be specified as a table argument.

The path selector name is 'service-time'.

Table parameters for each path: [<repeat_count> [<relative_throughput>]]
        <repeat_count>: The number of I/Os to dispatch using the selected
                        path before switching to the next path.
                        If not given, internal default is used. To check
                        the default value, see the activated table.
        <relative_throughput>: The relative throughput value of the path
                        among all paths in the path-group.
                        The valid range is 0-100.
                        If not given, minimum value '1' is used.
                        If '0' is given, the path isn't selected while
                        other paths having a positive value are available.

Status for each path: <status> <fail-count> <in-flight-size> \
                      <relative_throughput>
        <status>: 'A' if the path is active, 'F' if the path is failed.
        <fail-count>: The number of path failures.
        <in-flight-size>: The size of in-flight I/Os on the path.
        <relative_throughput>: The relative throughput value of the path
                        among all paths in the path-group.

Algorithm
=========

dm-service-time adds the I/O size to 'in-flight-size' when the I/O is
dispatched and substracts when completed.
Basically, dm-service-time selects a path having minimum service time
which is calculated by:

        ('in-flight-size' + 'size-of-incoming-io') / 'relative_throughput'

However, some optimizations below are used to reduce the calculation
as much as possible.

        1. If the paths have the same 'relative_throughput', skip
           the division and just compare the 'in-flight-size'.

        2. If the paths have the same 'in-flight-size', skip the division
           and just compare the 'relative_throughput'.

        3. If some paths have non-zero 'relative_throughput' and others
           have zero 'relative_throughput', ignore those paths with zero

'relative_throughput'.

If such optimizations can't be applied, calculate service time, and
compare service time.
If calculated service time is equal, the path having maximum
'relative_throughput' may be better. So compare 'relative_throughput'
then.

Examples
========
In case that 2 paths (sda and sdb) are used with repeat_count == 128
and sda has an average throughput 1GB/s and sdb has 4GB/s,
'relative_throughput' value may be '1' for sda and '4' for sdb.

```
# echo "0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 1 8:16 128 4" \
  dmsetup create test
#
# dmsetup table
test: 0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 1 8:16 128 4
#
# dmsetup status
test: 0 10 multipath 2 0 0 0 1 1 E 0 2 2 8:0 A 0 0 1 8:16 A 0 0 4
```


Or '2' for sda and '8' for sdb would be also true.

```
# echo "0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 2 8:16 128 8" \
  dmsetup create test
#
# dmsetup table
test: 0 10 multipath 0 0 1 1 service-time 0 2 2 8:0 128 2 8:16 128 8
#
# dmsetup status
test: 0 10 multipath 2 0 0 0 1 1 E 0 2 2 8:0 A 0 0 2 8:16 A 0 0 8
```