# SELFLET OPTIMIZATION MODEL

**Selflets**

We model our system with a set $\mathcal{N}$ of `SelfLet`s and use the index $n$ to denote a single `SelfLet`.

`SelfLet`s can be active (i.e., servicing requests) or passive (the virtual machine/server hosting it is powered-off). We assume a one-to-one correspondence between a virtual machine and a `SelfLet`. We use the variable $y_n$ to denote an active `SelfLet`.

$$y_n = \begin{cases} 1 & \text{if } \texttt{SelfLet } n \text{ is active} \\ 0 & \text{otherwise} \end{cases}$$

**Service Placement:**

Each `SelfLet` acts as a container of services. $\mathcal{S}$ is the set of available services and we denote a single service with index $s$.

Services are distributed among `SelfLet`s and can have replicas (i.e., the same service is deployed over multiple `SelfLet`s. We formalize this aspect with the following decision variable:

$$y_{n,s} = \begin{cases} 1 & \text{if } \texttt{SelfLet } n \text{ is running service } s \\ 0 & \text{otherwise} \end{cases}$$

Each service is offered by at least one `SelfLet`:

$$\sum_{n \in \mathcal{N}} y_{n,s} >= 1 \,, \forall s \in \mathcal{S}$$

If `SelfLet` is *passive* then no services are placed on it:

$$\sum_{s \in \mathcal{S}} y_{n,s} \leq y_n \cdot |\mathcal{S}| \,, \forall n \in \mathcal{N}$$

**Behavior for service:**

Each service can be implemented by different behaviors, however each behavior implements exactly one service. We denote the set of behaviors implementing service $s$ with the letter $\mathcal{B}_s$.

We call *active behavior* the behavior of service $s$ in `SelfLet` $n$ that is actively servicing requests. The following variable defines the active behaviors in `SelfLet` $n$.

$$z_{n,b} = \begin{cases} 1 & \text{if behavior } b \text{ is an } \textit{active behavior} \text{ in } \texttt{SelfLet } n \\ 0 & \text{otherwise} \end{cases}$$

At any given time, it is unique the behavior implementing service $s$ in `SelfLet` $n$ (if $s$ is offered by the `SelfLet`):

$$\sum_{b \in \mathcal{B}_s} z_{n,b} = y_{n,s} \, , \forall s \in \mathcal{S}, n \in \mathcal{N}$$

**Request rate:**

Each service can be invoked in different ways and in different `SelfLet`s. We denote the overall request rate for service $s$ with $\Lambda_s$ (i.e. the amount that is sent from the "external world" to the `SelfLet`s).

$\Lambda_s^n$ is the decision variable representing the *direct* (i.e., from clients to `SelfLet`) request rate of `SelfLet` $n$ for service $s$. The following constraint holds:

$$\sum_{n \in \mathcal{N}} \Lambda_s^n = \Lambda_s \, , \forall s \in \mathcal{S}$$

Requests are directed only toward `SelfLet`s offering that service:

$$\Lambda_s^n \leq y_{n,s} \cdot \Lambda_s \, , \forall s \in \mathcal{S}, n \in \mathcal{N}$$

**Behavior structure:**

Behaviors are represented as state diagrams and we refer to them with index $b$. The symbol $\mathcal{I}_b$ represents the set of internal states of behavior $b$. Each state represents an invocation of a service and it is represented by vector $\mathbf{S}_b = (s_1, s_2, \ldots, s_{|\mathcal{I}_b|})$.

Some behaviors do not invoke external services and are executed locally. We call these behaviors *abilities*. The structure of ability behaviors is fixed to be a behavior with only a single state.

A behavior can have multiple execution paths depending on some factors (for example the input parameters used in the service request). We model this aspect with a transition probability from each state to another. The structure of behavior $b$ is thus represented by the set of tuples:

$$\mathcal{B}_b = \{(tgt\_state,\ src\_state,\ p) \mid tgt\_state, src\_state \in \mathcal{I}_b, 0 \leq p \leq 1\}$$

where *tgt_state* and *src_state* are transitions' target and source states respectively and $p$ is the probability to perform the transition.

**Service rate for behavior states**

For each state $i$ of a behavior $b$ in `SelfLet` $n$ we define the total amount of requests that is executed with the symbol $\lambda_{n,b,i}$.

The total amount of requests received by the first state of the behavior (i.e., $i = 1$) is fixed according to the following constraint:

$$\lambda_{n,b,1} = \phi_{n,b} * z_{n,b}$$

where $\phi_{n,b}$ is the *total* amount of requests that is received by behavior $b$ in `SelfLet` $n$. Differently from $\Lambda_n^s$, the term $\phi_{n,b}$ also includes contributions from local and remote redirects. Its definition is given later.

The total amount of requests received by all the other states of the behavior is defined using this recursive formula for $i > 1$:

$$\lambda_{n,b,i} = \sum_{\substack{(i_t,i_s,p)\in\mathcal{B}_b \wedge \\ i=i_t}} \lambda_{n,b,i_s} \cdot p$$

**State execution** If the service invoked by a given state of a behavior is locally available it can be executed directly by the `SelfLet` that is requiring it. Otherwise the `SelfLet` must redirect the request to another `SelfLet` that is offering it.

For this reason we use two binary variables to denote the fact that state $i$ of behavior $b$ at `SelfLet` $n$ is executed locally or remotely. In particular:

$$l_{n,b,i} = \begin{cases} 1 & \text{if state } i \text{ of behavior } b \text{ is executed locally} \\ 0 & \text{otherwise} \end{cases}$$

and

$$r_{n,b,i} = \begin{cases} 1 & \text{if state } i \text{ of behavior } b \text{ is executed remotely} \\ 0 & \text{otherwise} \end{cases}$$

We use two separate binary variables since for each state three are the possible configurations: (i) local execution, (ii) remote execution, and (iii) the behavior is not executed at all. As stated by the following constraint, a state cannot be executed both locally and remotely:

$$l_{n,b,i} + r_{n,b,i} = z_{n,b}$$

If behavior $b$ is an ability then it is executed locally:

$$l_{n,b,i} \geq z_{n,b} \cdot \mathit{ability}(b)$$

where $\mathit{ability}(b)$ is a binary function that returns 1 if and only if behavior $b$ is an ability.

If the service requested by state $i$ of behavior $b$ is in selflet $n$ is available in the local `SelfLet` then it is executed locally:

$$l_{n,b,i} \geq y_{n,s}$$

where $s$ is the service requested by $i$.

**State demand**

If a behavior state is executed - both locally or remotely - it consumes some CPU time (i.e., demand). In particular, if the state $i$ of behavior $b$ corresponds to an ability we assume to know its demand time and we call it $AD_{b,i}$ (ability demand).

If state $i$ is executed remotely there is a fixed amount of CPU demand that is consumed (for example due to network communication and parameters passing); we refer to the remote CPU demand with the term $RD$ (remote demand).

We can define the CPU demand $d_{n,b,i}$ consumed by each state $i$ of the behavior $b$ in `SelfLet` $n$ with the following constraint:

$$d_{n,b,i} = l_{n,b,i} \cdot \mathit{ability}(b) \cdot AD_{b,i} + r_{n,b,i} \cdot RD$$

**Utilization**

In this part we present the constraints which are needed to quantify the CPU utilization for states of behaviors and for the `SelfLet`s. The utilization of state $i$, of behavior $b$ in `SelfLet` $n$ can be computed using the following constraint:

$$u_{n,b,i} = d_{n,b,i} \cdot \lambda_{n,b,i}$$

State utilization is limited by 1 and we also want to avoid the saturation condition:

$$u_{n,b,i} < 1$$

Utilization for `SelfLet` $n$ is defined as the sum of the utilization of all states belonging to all behaviors executing in $n$.

$$U_n = \sum_{b \in \mathcal{B},\ i \in \mathcal{I}_b} u_{n,b,i}$$

Utilization of states is bounded by 1 and is zero for non active `SelfLet`s

$$U_n < y_n$$

**Service redirect** In this part we characterize the redirects the service perform from one `SelfLet` to another. We call $\gamma_{n,b,i,m}$ the amount of requests redirected from `SelfLet` $n$ to `SelfLet` $m$ from state $i$ of behavior $b$.

In particular, the amount of outgoing traffic (i.e., redirected requests) from state $i$ of behavior $i$ toward all `SelfLet`s is defined by the following constraint:

$$\sum_{m \in \mathcal{N}} \gamma_{n,b,i,m} = \lambda_{n,b,i} \cdot (1 - ability(b))\ ,\ \forall b \in \mathcal{B}, i \in \mathcal{I}_b, n \in \mathcal{N}$$

Redirect is zero if the other `SelfLet` is not offering the requested service:

$$\gamma_{n,b,i,m} \leq y_{m,s} * \Lambda_s$$

where $s$ is the service requested by state $i$ in behavior $b$.

**Total rate**

We define $\lambda^l_{n,b}$ as the amount of requests received by behavior $b$ in `SelfLet` $n$ due to local redirects (i.e., originating in other behaviors of the same `SelfLet`).

$$\lambda^l_{n,b_1} = z_{n,b_1} \cdot \sum_{\substack{b_2 \in \mathcal{B} \wedge b_2 \neq b_1 \wedge i \in \mathcal{I}_{b_2} \\ serviceOfState(i,b_2) = \\ serviceOfBehavior(b_1)}} z_{n,b2} \cdot l_{n,b_2,i} \cdot \lambda_{n,b_2,i}$$

Similarly, we define $\lambda^r_{n,b}$ as the amount of requests received by behavior $b$ in `SelfLet` $n$ due to remote redirects (i.e., originating from other `SelfLet`s).

$$\lambda^r_{n,b_1} = z_{n,b_1} \cdot \sum_{\substack{n \neq m \wedge b_2 \in \mathcal{B} \wedge i \in \mathcal{I}_{b_2} \\ serviceOfState(i,b_2) = \\ serviceOfBehavior(b_1)}} \gamma_{m,b_2,i,n}$$

The total request rate for behavior $b$ in `SelfLet` $n$ is obtained by summing the direct requests, local request and remote requests in the following way:

$$\phi_{n,b} = \lambda^l_{n,b} + \lambda^r_{n,b} + \Lambda^n_{serviceOf(b)}$$

**Response time**

In order to compute the response time of service offered by a `SelfLet` we basically sum the response time of each state composing the behavior that implements it. We denote the response time of each state $i$ of behavior $b$ in `SelfLet` $n$ with the variable $R_{n,b,i}$.

The response time of each state can be defined in three separate ways depending on how the service invoked by the state is executed. In particular, the state may refer to a service implemented by (i) local ability, a (ii) local complex behavior, and finally a (iii) behavior executed in a remote `SelfLet`.

The following formula characterize the state response time considering the previously described cases:

$$\begin{aligned} R_{n,b,i} = z_{b,n} \cdot [ & \\ & l_{n,b_2,i} \cdot (\frac{d_{n,b,i}}{1 - U_n} + (1 - ability(b)) \cdot R_{n,serviceOfState(b,i))}) + \\ & r_{n,b_2,i} \cdot (\sum_{m \in \mathcal{N} \wedge m \neq n} x_{n,b,i,m} \cdot R_{m,serviceOfState(b,i)}) \qquad ] \end{aligned}$$

Response time of a service $s$ in `SelfLet` $n$ is the sum of response times of each state belonging to the implementing behavior weighted by the probability of executing that state. More precisely:

$$R_{n,s} = \sum_{b \in \mathcal{B}_s} \sum_{i \in \mathcal{I}_b} R_{n,b,i}$$

**SLA** Service response time is bounded by a threshold $\overline{R_s}$ defining the service level agreement

$$R_{n,s} \leq \overline{R_s} \; \forall n \in \mathcal{N}, s \in \mathcal{S}$$

**Objective Function:**

The objective function is the minimization of the active `SelfLet`s

$$\min \sum_{n \in \mathcal{N}} y_n$$

**Symbol Table**

| Symbol | Description |
|---|---|
| $\mathcal{N}$ | Set of logical `SelfLet`s |
| $n$ | `SelfLet` index |
| $y_n$ | Active `SelfLet` variable |
| $\mathcal{S}$ | Set of services |
| $s$ | Service index |
| $y_{n,s}$ | Service $s$ placed in `SelfLet` $n$ |
| $\mathcal{B}$ | Set of all behaviors |
| $\mathcal{B}_s$ | Set of behaviors implementing service $s$ |
| $z_{n,b}$ | Behavior $b$ is active is `SelfLet` $n$ |
| $\Lambda_s$ | Direct requests for service $s$ |
| $\Lambda_s^n$ | Direct requests for service $s$ at `SelfLet` $n$ |
| $\mathcal{I}_b$ | Set of states for behavior $b$ |
| $\mathbf{S}_b$ | Vector of invoked services for states of behavior $b$ |
| $\mathcal{B}_b$ | Set of tuples $(i_1, i_2, p)$ defining behavior structure |
| $p$ | Probability to execute a behavior transition |
| $\lambda_{n,b,i}$ | Total amount of requests executed by state $i$ of behavior $b$ in `SelfLet` $n$ |
| $\phi_{n,b}$ | Total amount of requests received by behavior $b$ in `SelfLet` $n$ |
| $l_{n,b,i}$ | Binary var. to indicate that state $i$ of behavior $b$ in `SelfLet` $n$ is executed locally |
| $r_{n,b,i}$ | Binary var. to indicate that state $i$ of behavior $b$ in `SelfLet` $n$ is executed remotely |
| $ability(b)$ | Binary function indicating whether behavior $b$ is an ability |
| $AD_{b,i}$ | Demand for state $i$ of behavior $b$ |
| $RD$ | Remote demand |
| $u_{n,b,i}$ | Utilization for state $i$ of behavior $b$ in `SelfLet` $n$ |
| $U_n$ | Utilization for `SelfLet` $n$ |
| $\gamma_{n,b,i,m}$ | Amount of outgoing traffic from state $i$ of behavior $b$ in `SelfLet` $n$ toward `SelfLet` $m$ |
| $serviceOfBehavior(b)$ | Function returning the service implemented by behavior $b$ |
| $serviceOfState(i, b)$ | Function returning the service required by state $i$ of behavior $b$ |

TABLE 1. Table of symbols