

Manual de uso simulador de AntNet en RMASE:

Alcance:

Este documento describe cómo utilizar la plataforma de simulación para AntNet y sus variantes, desarrollada sobre RMASE (*Routing Modelling Application Simulation Environment*). El objetivo es poder definir los parámetros que se deseen del algoritmo original para realizar diferentes pruebas, añadir nuevas funcionalidades al algoritmo y ser capaz de definir la topología de la red a conveniencia.

Versiones

Esta implementación probó ser funcional en la versión de Matlab R2019a, y la versión de Prowler es la 1.25

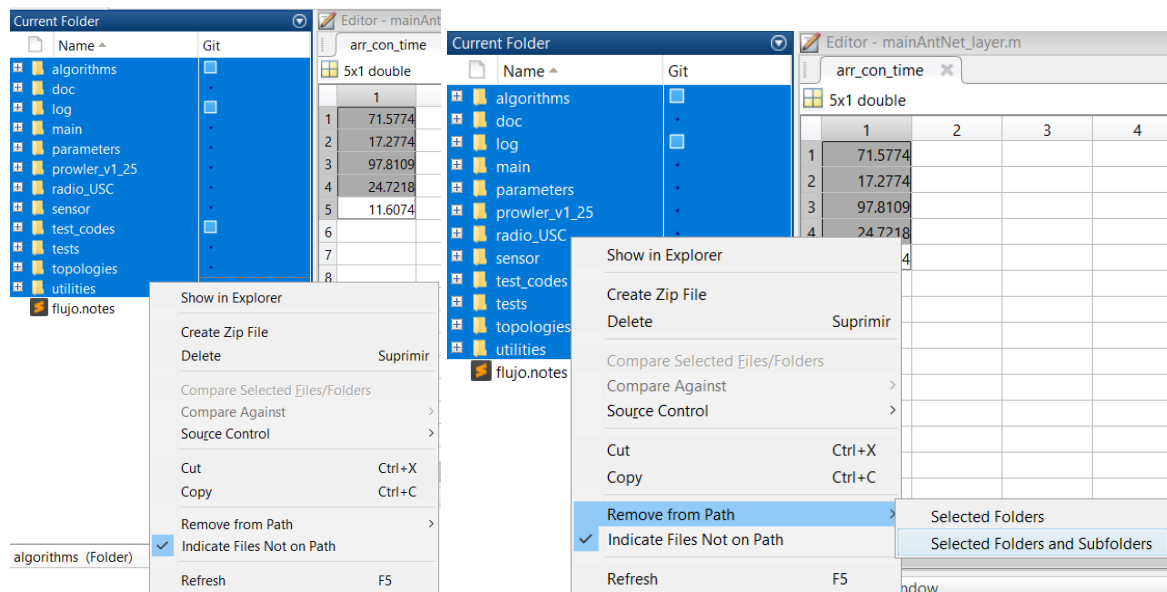
Documentación oficial

La documentación oficial de RMASE se encuentra como una página web en Rmase-1.1-share/doc/html/Rmase

Archivos necesarios

Todos los archivos necesarios para correr las pruebas se hallan en el directorio del repositorio: `Redes-de-Comunicacion-2021\Abraham Roquel\documentos\Rmase-1.1-share`

Para tener todas las dependencias necesarias es necesario añadir todos los folders y sub-folders del repositorio al directorio de trabajo por medio de los siguiente pasos:

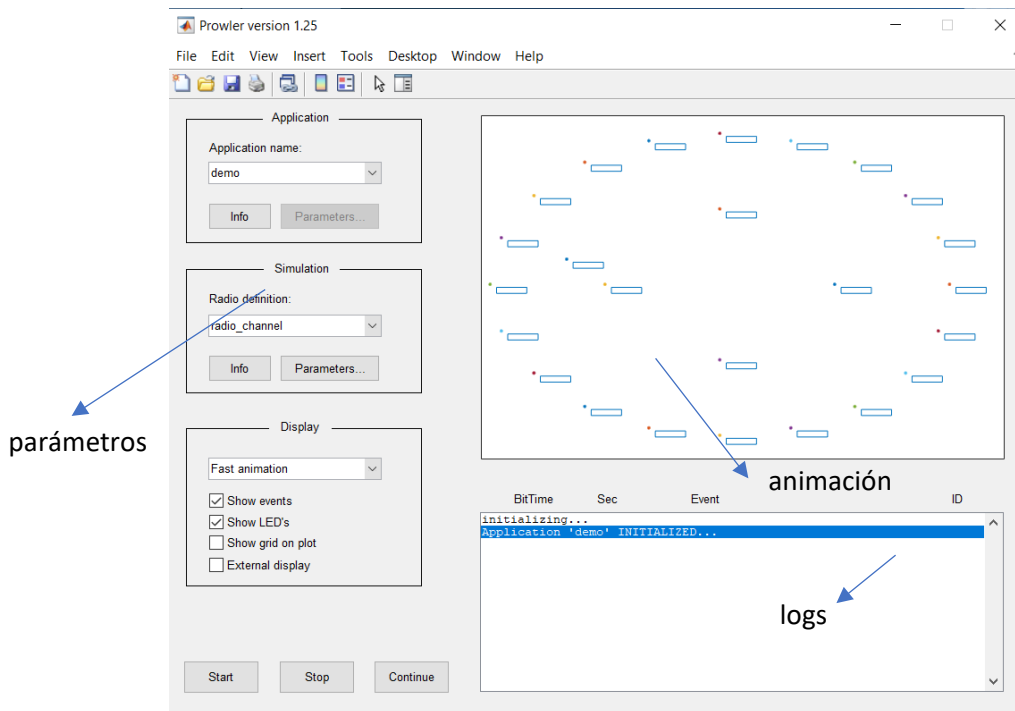


Esto también es posible ser realizado por medio de la línea de comandos, si `d` es el directorio con los archivos, entonces se tiene:

```
addpath(genpath('matlab/myfiles'))
```

Inicialización

Luego de haber incluido todos los archivos necesarios al directorio de trabajo, se puede ejecutar el comando `prowler` en la línea de comandos, luego de lo cual aparecerá lo siguiente:



En esta pantalla principal es posible elegir:

- La aplicación a usar: pueden explorarse las otras, pero la de interés es RMASE
- El modelo de radio: se pueden elegir parámetros como el threshold de potencia para definir que se recibe un paquete, la energía que esto consume, etc.
- Parámetros visuales, como la activación de una grilla para visualizar las coordenadas, o bien un display externo para ver en otra instancia de una figura la simulación como tal.

Parámetros generales de RMASE

Topología

Al elegir la aplicación de RMASE y hacer click sobre el botón de selección de parámetros se obtiene lo que se muestra en la figura a continuación. En esta parte se configura la topología a utilizar, en caso se quiera usar una cuadrícula, por medio de diversas opciones. Para más información ver Rmase-1.1-share/doc/html/Rmase/models.htm en la sección de topology model. También es posible usar un archivo (ver comentarios de `topologies/test_topo.m`)

Prowler - Application Parameters

Parameters for application 'RMASE'

Parameter Groups = Network Topology

Xsize = 7

Ysize = 7

Xdist = 1

Ydist = 1

Xoffset = 0

Yoffset = 0

Xdensity = 1

Ydensity = 1

Xshift = 0

Yshift = 0

☐ wraparound

AliveProb = 1

☐ UseTopologyFile

TopologyFileName = none

Default Apply Close Cancel

Nodo fuente y destino

El nodo fuente que empezará a generar paquetes se encuentra en la sección de Source, donde se pueden especificar parámetros sobre los cuales RMASE elegirá de manera aleatoria la fuente. Algo muy similar se puede especificar en la sección de destination.

Parameters for application 'RMASE'

Parameter Groups = Application Source

SourceType = static

SourceCenterType = random

SourceCenterX = 0

SourceCenterY = 0

SourceRadius = 1

SourcePercentage = 1

☒ SourceUnique

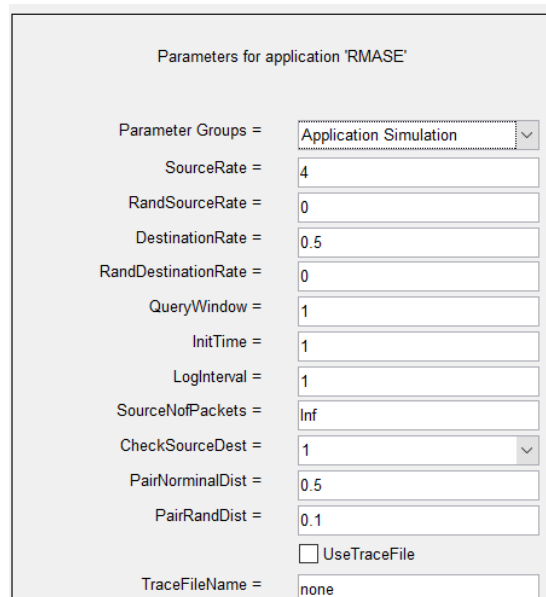
SourceSpeedX = -0.2

SourceSpeedY = -0.2

RandSourceSpeed = 0.1

Otros parámetros generales

En la sección de *simulation* se puede elegir la tasa de transmisión a nivel de capa de aplicación que tendrá el nodo fuente, también el nodo destino. **Para las pruebas a realizar con la implementación de AntNet hecha, es necesario poner el SourceRate en cero para no crear conflicto con los paquetes de ruteo.**



Parameters for application 'RMASE'

Parameter Groups = Application Simulation

SourceRate = 4

RandSourceRate = 0

DestinationRate = 0.5

RandDestinationRate = 0

QueryWindow = 1

InitTime = 1

LogInterval = 1

SourceNoPackets = Inf

CheckSourceDest = 1

PairNominalDist = 0.5

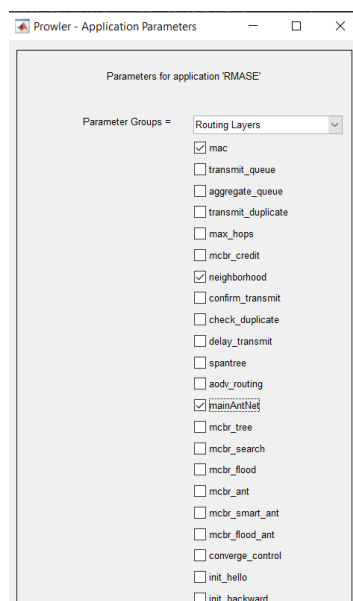
PairRandDist = 0.1

☐ UseTraceFile

TraceFileName = none

Elegir las capas a utilizar

En la sección de layers aparece un menú como el que se muestra, donde se pueden seleccionar las capas a utilizar. **Para fines de correr el código es necesario elegir la capa mac, de neighborhood y mainAntNet** (si se crean otras luego también es posible añadirlas al menú).



Prowler - Application Parameters

Parameters for application 'RMASE'

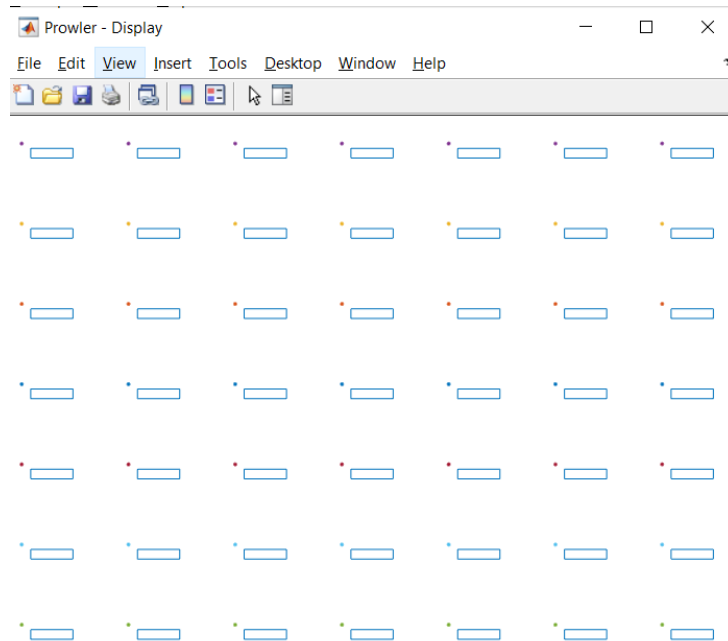
Parameter Groups = Routing Layers

- ☒ mac
- ☐ transmit_queue
- ☐ aggregate_queue
- ☐ transmit_duplicate
- ☐ max_hops
- ☐ mcb_r_credit
- ☒ neighborhood
- ☐ confirm_transmit
- ☐ check_duplicate
- ☐ delay_transmit
- ☐ spantree
- ☐ aodv_routing
- ☒ mainAntNet
- ☐ mcb_r_tree
- ☐ mcb_r_search
- ☐ mcb_r_flood
- ☐ mcb_r_ant
- ☐ mcb_r_smart_ant
- ☐ mcb_r_flood_ant
- ☐ converge_control
- ☐ init_hello
- ☐ init_backward

Luego de elegir los parámetros deseados se presiona Apply, luego close.

Tipo de presentación

Para que recortes de pantalla de la simulación se guarden automáticamente es necesario elegir la opción de *External display*, la cual generará otra figura:

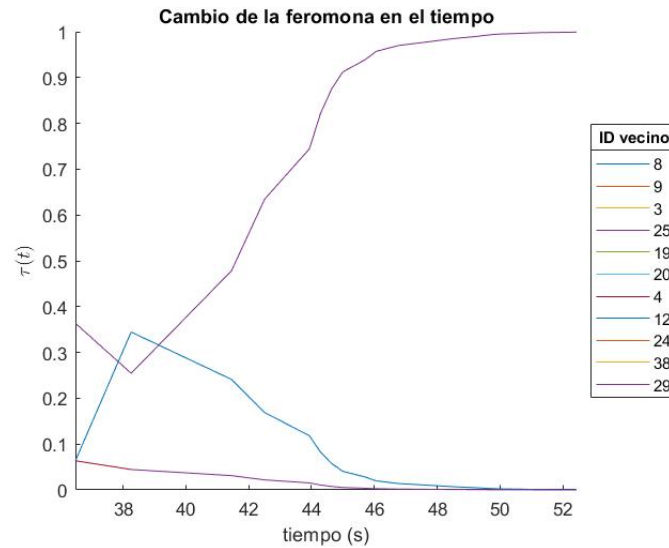


Ejecutar la simulación

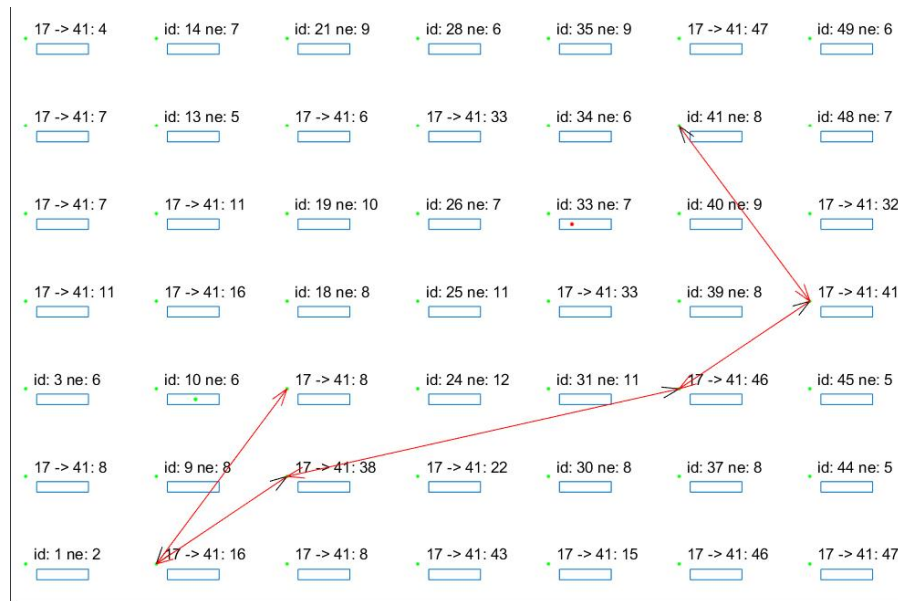
Con la topología y demás parámetros seleccionados basta con presionar Start, luego de lo cual se tendrá la animación y se guardarán los resultados (en figuras y un archivo mat de resultados). Los resultados almacenados son:

- Una captura de pantalla por cada vez que una hormiga de backward regresa al nodo fuente
- Un archivo mat con la evolución de las feromonas en el tiempo. Para mostrar la figura basta con ejecutar la función `abe_myfun()`
- Un archivo mat al que se añaden los resultados de retardo de arranque, retardo de convergencia y RTT de convergencia (son 3 arreglos, análogo a una operación de `append()` para luego analizar los resultados de varias corridas en bloque).

Este es un ejemplo de una figura de feromona generada:



Estos son ejemplos de las figuras guardadas de forma automática:



Parámetros configurables de la implementación de AntNet

Todos los archivos relevantes desarrollados para esta implementación están en el directorio `Rmase-1.1-share\algorithms\custom_components\antnet`

- `mainAntNet_Layer`: es la porción principal de código donde está la mayoría de la funcionalidad del proyecto
- `CStack`: es un archivo que define una clase de pilas, que son útiles en el algoritmo para almacenar el camino de los paquetes en la red.
- `new_layer_template`: es una plantilla sencilla para escribir un nuevo componente de ruteo

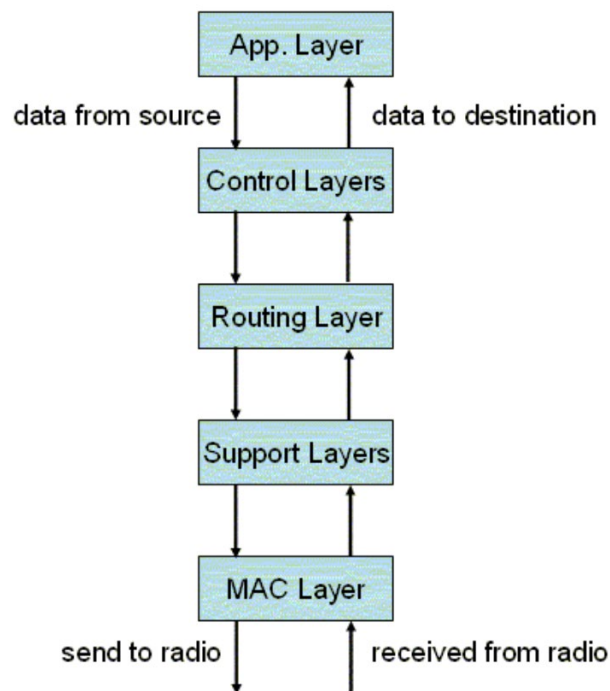
mainAntNet_Layer:

- Variables configurables (se cambian en el case de 'Init_Application')
 - forward_ant_interval: en bit-time (con 1 bit-time = 0.00425 s) el tiempo para esperar entre envíos consecutivos de hormigas de forward
 - Tp: cantidad de slots en modo de descubrimiento de vecinos
 - TmaxHops: cantidad máxima de saltos de hormiga de forward
 - force_endpoints: determina si la fuente y el destino los determina RMASE o se determinan por medio de las variables forced_source y forced_destination
 - window_size: número de muestras sobre el cual calcular el mejor valor histórico de RTT
 - stop_conditon: porcentaje de error de la solución antes de converger
 - reinf_mode: determina qué modo de refuerzo usar (1 para el libro de Gianno, 2 para solo considerar Wbest, 3 para considerar solo la media)
 - penalize_forward_loss: determina si se penalizan o no las pérdidas de forward
 - Los hiperparámetros de AntNet que se describen en el trabajo de tesis en la sección de caracterización de AntNet es pueden cambiar con las variables: p_zeta, p_c, p_v, p_c1, p_c2
 - El color de las flechas: por medio de las variables CUSTOM_COLOR_FORWARD y CUSTOM_COLOR_BACKWARD se pueden cambiar el color de las flechas de recepción de hormigas de forward y de backward como un vector fila que es una triada de código RGB.
- Porciones de código configurables:
 - Calcular el modelo de la red en función del rtt: la función udpate_and_get_r debe devolver una nueva matriz de modelos, de RTTs históricos y el refuerzo r. Dentro de ella se pueden modificar las líneas correspondientes para asignar a la variable new_S(1, target_destination_index) el nuevo valor de la media y a new_S(2, target_destination_index) el nuevo valor de la varianza.
 - El refuerzo de backward: la función get_reinforcement debe devolver el refuerzo r con $1 > r > 0$ como función de la media, varianza, mejor valor, valor actual de RTT y los hiperparámetros de antnet. No es necesario usar todos los valores, se puede usar cualquier regla de asignación para r.

Cómo añadir un nuevo algoritmo/componente de ruteo

Se debe crear un nuevo archivo siguiendo la plantilla en `new_layer_template`. Los eventos vienen de capas inferiores a superiores y los comandos en sentido contrario. Se debe definir el código para cada evento y comando de interés. Para añadirlo a la lista de capas disponibles se debe añadir una línea al archivo `all_apps_layers.m` con el nombre del archivo (sin el sufijo `layer`)

```
'ant_routing', ...  
'mainAntNet', ... % adición de la nueva capa definida  
'mabr_tree', ...
```



- Las variables de tipo PERSISTENT mantienen su valor luego de cada iteración a menos que se les modifique
- Las variables de tipo GLOBAL mantienen su valor a menos que se modifiquen y son accesibles por medio de cualquier componente de ruteo.
- NO HAY VARIABLES COMPARTIDAS ENTRE DIFERENTES AGENTES.