Projekt ODOKRIM

SYMULATOR SIECI RADIOWEJ

PODSUMOWANIE

Spis treści

1. Wprowadzenie

Niniejszy dokument stanowi opis symulatora, który powstał w ramach projektu ODOKRIM.

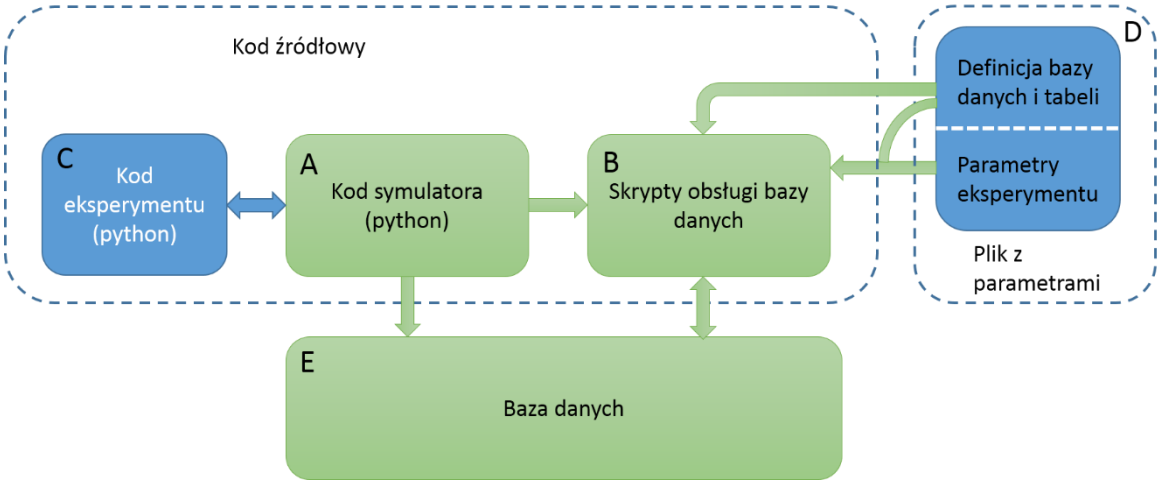
Omawiany symulator umożliwia badanie predefiniowanych algorytmów jak i nowych komponentów wprowadzanych przez użytkownika. Modeluje on sieć radiową składającą się z dowolnej liczby węzłów. Jest to symulator typu kwazi-statycznego, co oznacza, że ruch obiektów jest w nim modelowany poprzez zmienność środowiska radiowego (zaniki szybkie).

Symulator działa w oparciu o dynamiczny dziennik zdarzeń. Oznacza to, że symulator przechowuje w pamięci dyskretny ciąg zdarzeń, z których każde zostanie wykonane w określonym momencie czasu. Dynamiczność oznacza możliwość manipulowania dziennikiem, w szczególności dodawanie zdarzeń w zależności od wykonania innych zdarzeń. W odróżnieniu od symulatorów ciągłych, relacja czasowa pomiędzy zdarzeniami nie musi być stała.

Symulator został napisany w języku Python 2.7, wykorzystuje środowisko bazodanowe. Możliwe jest korzystanie z baz typu SQLite, PostgreSQL oraz MySQL, przy czym ta ostatnia jest zalecana.

2. Elementy składowe symulatora

Symulator składa się z kilku elementów tworzących wspólne środowisko. Poniższy schemat obrazuje części składowe środowiska symulacyjnego wraz z zależnościami pomiędzy poszczególnymi elementami. Kolorem niebieskim oznaczono elementy, które użytkownik może zmieniać, natomiast zielonym te, które nie powinny być zmieniane przez użytkownika.



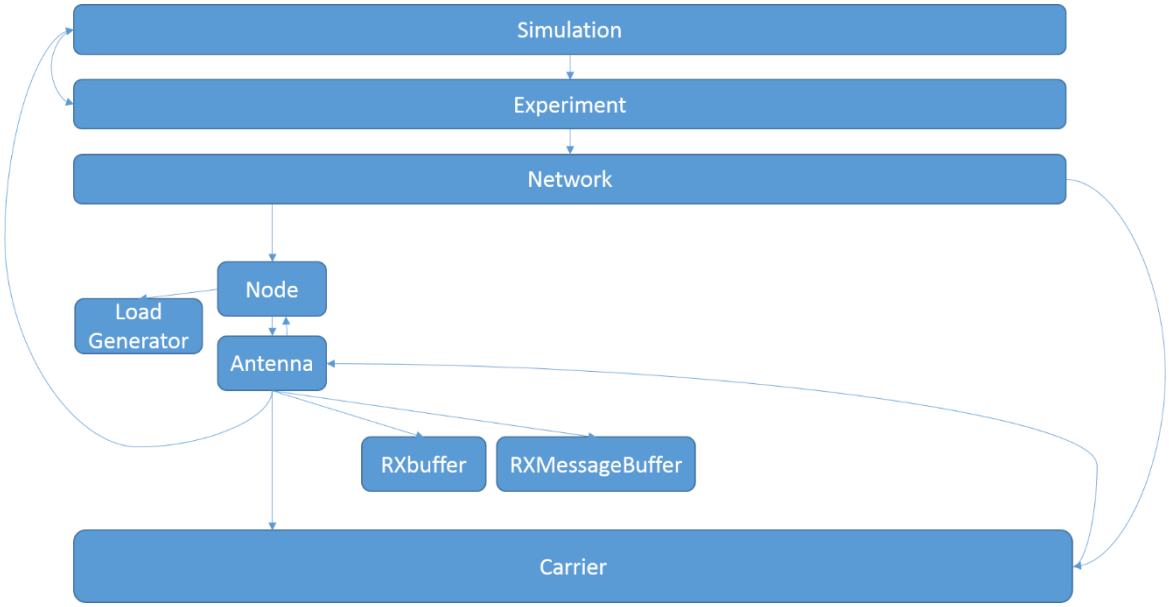
RYS. 1 – SCHEMAT ŚRODOWISKA SYMULACYJNEGO WRAZ Z ZALEŻNOŚCIAMI POMIĘDZY POSZCZEGÓLNYMI ELEMENTAMI

W symulatorze można wyróżnić następujące najważniejsze elementy:

- Kod symulatora (moduł A na rysunku 1) - niezmienny element środowiska symulacyjnego. Składa się z modułów python, które modelują sieć radiową. API kodu symulatora opisano w rozdziale 3.
- Baza danych (moduł E) – pełni rolę historii uruchomionych eksperymentów. Przechowywane w niej są parametry poszczególnych eksperymentów (wejście symulacji) oraz wiadomości generowane przez symulator w trakcie symulacji (wyjście symulacji). Pozwala to na późniejsza analizę wyników symulacji (na przykład wyliczenie podsumowujących statystyk) i powiązanie ich z parametrami wejściowymi. Bazę danych obsługują skrypty, będące częścią modułu B, napisane w języku python.
- Kod eksperymentu – zmienny element symulatora, definiowany przez użytkownika. Składa się z opisu parametrów wejściowych symulacji (moduł D - plik zapisany w języku json) oraz opisu przebiegu eksperymentu (moduł C – plik python, w którym użytkownik definiuje zachowanie poszczególnych elementów sieci oraz może wprowadzać własne algorytmy).

3. Struktura oprogramowania

Struktura najważniejszych bloków funkcjonalnych symulatora została przedstawiona na poniższym schemacie.



RYS. 2 - STRUKTURA OPROGRAMOWANIA

Obiekt klasy Simulation zawiera m.in. dziennik zdarzeń, aktualny czas oraz przechowuje obiekt klasy Experiment. Obiekt Experiment zawiera parametry konfiguracyjne. Obiekt klasy Network, który jest zapisany w obiekcie Experiment, jest kontenerem, w którym, w liście nodes, przechowywane są wszystkie węzły radiowe symulacji (obiekty klasy Node). Każdy „node" może mieć przypisany generator zdarzeń (Load Generator), który, zgodnie z określonym rozkładem, będzie wytwarzał zadane zdarzenia. „Node" ma też przypisaną antenę, która pełni rolę łącznika między węzłem a medium transmisyjnym. Antena, w obiektach RXBuffer i RXMessageBuffer przechowuje też informacje o stanie kanału radiowego w czasie poszczególnych transmisji oraz odebrane wiadomości. Obiekt Carrier modeluje medium transmisyjne. Realizuje rozsyłanie transmitowanych wiadomości wraz z modelowaniem zjawisk zachodzących w kanale radiowym.

Poniżej opisano najważniejsze elementy wpływające na modelowanie kanału radiowego.

3.1. Rozmieszczenie węzłów radiowych

Węzły mogą zostać rozmieszczone losowo w prostokącie o zadanym wymiarze lub też losowo na linii o zadanej długości.

3.2. Zaniki szybkie

W symulatorze zaimplementowano zaniki szybkie. Moduł obsługuje dwa rodzaje zaników: płaskie, a więc de facto kanał bez zaników, oraz zaniki zgodne z rozkładem Rayleigha.

Zaniki Rayleigha generowane są zgodnie z modelem zaproponowanym w [1]. Jest to realizacja klasycznego modelu Jakes'a, w którym zaniki szybkie modelowane są poprzez sumę wielu przesuniętych w fazie sinusoid. W zaimplementowanym modelu użyto dwudziestu oscylatorów generujących fale sinusoidalne. Modelowanie zaników odbywa się wg poniższych wzorów [5].

$$\bar{u}(t) = \bar{u}_c(t) + j\bar{u}_s(t)$$

$$\bar{u}_c(t) = \frac{2}{\sqrt{N}} \sum_{n=1}^{M+1} a_n \cos(\omega_n t)$$

$$\bar{u}_s(t) = \frac{2}{\sqrt{N}} \sum_{n=1}^{M+1} b_n \cos(\omega_n t)$$

, gdzie:

$$N = 4M + 2$$

$$\omega_n = 2\pi f_d$$

$$a_n = \begin{cases} 2cos\beta_n, & n = 1,2,\dots,M \\ \sqrt{2}\cos\beta_{M+1}, & n = M+1 \end{cases}$$

$$b_n = \begin{cases} 2sin\beta_n, & n = 1,2,\dots,M \\ \sqrt{2}\sin\beta_{M+1}, & n = M+1 \end{cases}$$

$$\beta_n = \begin{cases} \frac{\pi n}{M}, & n = 1,2,\dots,M \\ \frac{\pi}{4}, & n = M+1 \end{cases}$$

$$\omega_n = \begin{cases} \omega_d cos\frac{2\pi n}{N}, & n = 1,2,\dots,M \\ \omega_d, & n = M+1 \end{cases}$$

f_d to częstotliwość Dopplera wynikająca z ruchu obiektu, a M to liczba oscylatorów użytych w symulatorze.

3.3. Tłumienie sygnału radiowego

Moduł odpowiada za modelowanie tłumienia trasy radiowej. Zaimplementowano dwa modele tłumienia:

- Indoor Channel model, opisany w [3], w którym tłumienie wyliczane jest wg wzoru:

$$L = 15{,}3 + 37{,}6 * log_{10}(R)$$

Gdzie R to dystans w metrach

- Extended Hata Model, opisany w [4], w którym tłumienie wyliczane jest wg poniższej tabeli.

TABELA 1 - TŁUMIENIE KANAŁU RADIOWEGO WG MODELU EXTENDED-HATA

Odległość(d)	**Typ obszaru (areaType)**	**Częstotliwość (f), MHz**	**Tłumienie (L)**
< 0,04 km			$L = 32{,}4 + 20 \cdot \log_{10}(f) + 10 \cdot log_{10}\left(d^2 + \frac{(h_1 - h_2)^2}{10^6}\right)$
>0,1 km	Urban	$30 < f \leq 150$	$L = 69{,}6 + 26{,}2 \cdot \log_{10}(150) - 20 \cdot \log_{10}\left(\frac{150}{f}\right) - 13{,}82 \cdot \log_{10}(\max\{30;\ h_1\}) - a - b$
		$150 < f \leq 1500$	$L = 69{,}6 + 26{,}2 \cdot \log_{10}(f) - 13{,}82 \cdot \log_{10}(\max\{30;\ h_1\}) - a - b$
		$1500 < f \leq 2000$	$L = 46{,}3 + 33{,}9 \cdot \log_{10}(f) - 13{,}82 \cdot \log_{10}(\max\{30;\ h_1\}) - a - b$
		$2000 < f \leq 3000$	$L = 46{,}3 + 33{,}9 \cdot \log_{10}(2000) + 10 \cdot \log_{10}\left(\frac{f}{2000}\right) - 13{,}82 \cdot \log_{10}(\max\{30;\ h_1\}) - a - b$
	Suburban		$L = -2\log_{10}(\frac{min(max(150, f_{MHz}), 2000)}{28})^2 - 5{,}4$
	Rural		$L = -4.78\ \log_{10}(min(max(150, f\_MHz), 2000))^2 + 18.33\ \log_{10}(min(max(150, f\_MHz), 2000)) - 40.94$
0,04 – 0,1 km			$L(d) = L(0{,}04) + \left(\frac{\log_{10}(d) - \log_{10}(0{,}04)}{\log_{10}(0{,}1) - \log_{10}(0{,}04)}\right) \cdot \big(L(0{,}10) - L(0{,}04)\big)$

Gdzie h_1 i h_2 to wysokości, na jakiej znajdują się anteny dwóch rozpatrywanych węzłów,

$$a = (1{,}1 \cdot \log_{10}(f) - 0{,}7) \cdot \min\{10;\ h_2\} - (1{,}56 \cdot \log_{10}(f) - 0{,}8) + \max\left\{0;\ \ 20 \cdot \log_{10}\left(\frac{h_2}{10}\right)\right\},$$

$$b = \min\left\{0;\ \ 20 \cdot \log_{10}\left(\frac{h_1}{30}\right)\right\}.$$

4. Parametry symulatora

W poniższej tabeli zaprezentowane parametry, które użytkownik może określić za pomocą zewnętrznego pliku z parametrami. Część z tych parametrów może podlegać dynamicznym zmianom w trakcie symulacji. Dobrym przykładem jest np. moc nadawcza.

TABELA 2 - PARAMETRY SYMULATORA

Nazwa parametru	Znaczenie
circle	
a	parametry algorytmu zliczania węzłów
max_n	
circle_repeats	
beep	
p	parametry algorytmu identyfikacji węzłów
T	
active_nodes	
T_repeats	
window	
csma	
delta	parametry algorytmu CSMA
k	
lambda	
number_of_messages	
estimated_n	
generator_type	
generator_mean	
relay	

std_packet_delay	parametry algorytmu transmisji multi-hop
alarm_packet_delay	
generator_mean	
alarm_probability	
resource_blocks	
number_of_messages	
buffer_state_metric	
relay_selection	
convergecast	
strategy	parametru algorytmu trasowania
},	
rbo	
tKey	
tData	
number_of_messages	
n	liczba węzłów
layout_type	sposób rozmieszczenia węzłów
layout_xrange_max	wymiary środowiska symulacyjnego
layout_xrange_min	
layout_yrange_max	
layout_yrange_min	
areaType	typ obszaru symulacji
receiverSensitivity	czułość odbiornika
transmitPower	moc transmisyjna
frequency	częstotliwość fali nośnej
velocity	prędkość węzłów
thermal_noise	szum termiczny
fast_fading_type	typ zaników szybkich
pathloss_type	typ modelu tłumienia
SNRthreshold	progowa wartość SNR
max_time	maksymalny czas symulacji
randSeed	ziarno generatora losowego

5. Cechy symulatora

Symulator stworzony w ramach projektu ODOKRIM posiada szereg cech, które ułatwiają używanie symulatora nie tylko przez osoby bezpośrednio zaangażowane w prace programistyczne, ale także przez dowolnego członka zespołu badawczego po krótkim przeszkoleniu lub na podstawie zawartych w symulatorze przykładów.

5.1. Wielowątkowość

Symulator został napisany w taki sposób, że możliwe jest uruchomienie, za pomocą jednej komendy, szeregu równoległych symulacji. Znacznie przyspiesza to zarówno czas przygotowania eksperymentów, jak i czas ich wykonywania. Symulator analizuje zestaw przekazanych mu parametrów eksperymentów, a następnie każdą symulację uruchamia z inna kombinacją parametrów. Jest to szczególnie przydatne do modelowania eksperymentów metodą Monte Carlo, gdzie jednocześnie toczą się symulacje o takich samych parametrach, w których generator losowy został zainicjowany inną liczbą. W szczególności oznacza to, że możliwe jest uzyskanie wiarygodnych statystycznie wyników w czasie wykonywania jednej symulacji.

5.2. API

W symulatorze stworzono także API (Application Programming Interface) umożliwiające zaprogramowanie nowego algorytmu z zestawu narzędzi istniejących w symulatorze. Użytkownik nie znający całości kodu symulatora może w prosty sposób określić środowisko symulacyjne, manipulować parametrami radiowymi węzłów a także dodawać własne algorytmy modyfikujące lub rozszerzające zachowanie węzłów radiowych. Dzięki temu użytkownik może skupić się na dodawaniu algorytmów wewnętrznych węzłów bez konieczności zmian w samym symulatorze.

6. Literatura

[1] Jakes, W. C. (1974). Microwave Mobile Communications.

[2] Marius F. Pop, N. C. (2001). Limitations of Sum-of-Sinusoids Fading Channel. IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. 49, NO. 4, APRIL 2001

[3] 3GPP. TR 36.814 v 900 Evolved Universal Terrestrial Radio Access (E-UTRA); Further advancements for E-UTRA physical layer aspects.

[4] European Communications Office. SEAMCAT Handbook, 2010

[5] Yahong Rosa Zheng, Chengshan Xiao. Simulation Models With Correct Statistical Properties for Rayleigh Fading Channels. IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. 51, NO. 6, JUNE 2003

7. Pozostałe informacje

Symulator powstał dzięki dofinansowaniu z Narodowego Centrum Badań i Rozwoju w ramach projektu „Optymalizacja dostępu do kanału radiowego w mikrosieciach” (ODOKRIM).