Henryk KRAWCZYK[1,2], Piotr ORZECHOWSKI[2]
Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki[1],
Centrum Informatyczne Trójmiejskiej Akademickiej Sieci Komputerowej[2]

INTELIGENTNE ZARZĄDZANIE USŁUGAMI CHMUROWYMI

Streszczenie. Rozwój chmur obliczeniowych stanowi wyzwanie dla nowych efektywnych metod zarządzania zasobami chmurowymi, zwłaszcza, że oprócz usług typu SaaS rozwija się nowe kategorie usług jak obliczenia brzegowe czy wielochmurowe. W pracy zaproponowano ogólny model zarządzania usługami oraz efektywne procedury alokacji zasobów. Podkreślono potrzebę oszacowania parametrów zasobów chmury by zapewnić wykonanie żądanych usług. Przedstawiono również przykładowe rozwiązanie bazujące na sztucznej inteligencji, wdrożone w chmurze TASKcloud rozwijanej w CI TASK.

INTELIGENT MANAGEMENT OF CLOUD SERVICES

Summary. Rapid development of cloud computing technologies increases needs for new efficient cloud management strategies. It is evident not only for SaaS solutions but also for edge and multicloud computing. In the paper service-oriented management model is proposed and efficient resource allocation procedures are considered. The most important is to estimate correctly parameters of cloud resources in order to achieve execution of the required services. The example of such solution based on artificial intelligence deployed in TASKcloud developed in CI TASK is also given.

1. Wprowadzenie

Chmura obliczeniowa stanowi charakterystyczną infrastrukturę informatyczną, której zasoby są dostępne na żądanie dla jej użytkowników w formie usług. Wyróżnia się trzy podstawowe modele dostępu [11]:

- IaaS (*Infrastructure as a Service*) – platformy zapewniającej dostęp do infrastruktury sprzętowej określonej poprzez zasoby obliczeniowe, pamięciowe, komunikacyjne oraz dyskowe,
- PaaS (*Platform as a Service*) – platformy, która oferuje dostęp do systemów wytwarzania i wdrożenia usług i aplikacji IT,
- SaaS (*Software as a Service*) – platformy oferującej możliwości skorzystania z gotowych aplikacji i usług wdrożonych na infrastrukturze chmurowej.

W związku z powyższym wyróżniamy też trzech różnych aktorów związanych z funkcjonowaniem chmury obliczeniowej. Jedni to dostawcy chmury obliczeniowej (*cloud providers*), którzy zarządzają centrami danych i oferują wirtualne zasoby chmury dla różnych użytkowników. Drugą grupę stanowią użytkownicy chmury (*cloud users*), którzy wypożyczają od dostawców chmury zasoby obliczeniowe, takie jak np. maszyny wirtualne (*virtual machines*) i wdrażają na nich własne usługi i aplikacje, następnie udostępniane w modelu SaaS. Użytkownicy chmury na potrzeby wdrażania usług mogą też wykorzystywać technologie konteneryzacji i dostępne oprogramowanie chmury (PaaS). Tak przygotowane usługi i aplikacje dostarczane są użytkownikom końcowym (*end users*). W dalszej części artykułu użytkownicy końcowi nazywani będą użytkownikami usług (ogólnie usług lub aplikacji chmurowych). Tacy użytkownicy poprzez swoje żądania generują główne obciążenie chmury poprzez wywołanie instancji usług czy aplikacji, tzw. zadania (*workloads*) [16], które są przetwarzane na zasobach wskazanych przez system zarządzania. Tego typu użytkownicy nie odgrywają istotnej roli w zarządzaniu chmurą, ale ich oczekiwania i zachowania wpływają na decyzje zarządzające z punktu widzenia np. atrakcyjności i wysokiej jakości wykorzystanych przez nich usług czy aplikacji. Natomiast dostawcy chmury koncentrują się przede wszystkim na zarządzaniu podstawowymi zasobami fizycznymi dostępnymi dla różnego typu użytkowników w formie zasobów wirtualnych.

Przez usługę czy aplikację rozumieć będziemy wdrożoną sekwencję kodu źródłowego w określonym języku programowania lub odpowiadający im kod wykonywalny w danym środowisku obliczeniowym. Każde wywołanie tego kodu otrzymuje własną instancję, ogólnie nazywaną zadaniem, która powoduje odpowiednie obciążenie zasobów fizycznych (tzw. *workloads*). W przypadku chmur obliczeniowych wyróżnia się obrazy, które są statycznym zbiorem wykonywalnego kodu, który może być wdrożony bezpośrednio na maszynie wirtualnej bądź może być opakowany w kontener, dzięki czemu może być uruchamiany w dowolnym środowisku chmurowym wspierającym konteneryzację. Kontener zawiera wszystkie niezbędne elementy niezbędne do poprawnego wykonania osadzonej w nim aplikacji wykorzystując jednocześnie jądro systemu operacyjnego hosta (gościa). Inaczej jest w przypadku maszyny wirtualnej, gdzie obraz maszyny zawiera system operacyjny wraz z wszystkimi bibliotekami.

Wycena zasobów chmurowych związana jest bezpośrednio z czasem ich wykonania oraz z wykorzystanym rozmiarem zasobów obliczeniowych i danych. Dostawca chmury monitoruje wielkości faktycznie zarezerwowanych i wykorzystywanych zasobów fizycznych na poziomie IaaS zarówno przez użytkowników chmury, jak i użytkowników aplikacji. Poza tym użytkownicy chmury mogą czerpać profity od użytkowników końcowych na podstawie wykorzystania ich usług i aplikacji na poziomie SaaS. Taka wycena podawana jest w umowach i dotyczyć może miesięcznej opłaty za wykorzystanie odpowiednich zasobów wirtualnych, uwzględniając również typ wykorzystywanych usług i aplikacji, czy przyjęty poziomu dostępności oraz jakości usług, tzw. SLA (*Service Level Agreement*) [14]. Zawarta umowa może też uwzględniać pewne odszkodowania związane z naruszeniem uzgodnionych wymagań (np. dłuższego czasu przestoju chmury).

Dużym wyzwaniem jest więc zapewnienie elastyczności środowiska chmurowego minimalizującego koszty eksploatacyjne i operacyjne chmur obliczeniowych [3]. Cecha

ta powinna nie tylko dotyczyć pojedynczych klientów, ale również bądź przede wszystkim wielu różnych użytkowników realizujących jednocześnie swoje usługi chmurowe. Co więcej powinna uwzględniać dostępną architekturę chmury, która obecnie wykorzystuje wirtualizację na poziomie maszyn wirtualnych bądź kontenerów [12,8]. Poza tym oprócz typowych rozwiązań chmurowych, wyróżnić można inne rozwiązania jak chmura brzegowa związana z wykorzystaniem różnego typu urządzeń cyfrowych (Internetu Rzeczy - *edge computing*) [2], czy kooperacja wielu chmur obliczeniowych (*multicloud*) [5]. Zapewnienie elastyczności zarządzania zasobami chmur obliczeniowych polega więc na dynamicznym dostosowaniu oferowanych zasobów chmury do bieżących potrzeb użytkowników. Jest to dużym wyzwaniem z uwagi na zmienność żądań klientów i trudność przewidzenia wielkości zasobów (profilowanie) niezbędnych do realizacji takich żądań. Na ogół estymuje się te wielkości na podstawie obserwacji żądań użytkowników i wykorzystywanych przez nich zasobów w określonym przedziale czasu [6]. Co więcej wyróżnia się różnego typu podejścia zapewniające uzyskanie wymaganej elastyczności. Do najważniejszych z nich należy:

- przydział (alokacja) (*distribution*) zasobów dla żądań użytkowników pojawiających się w zadanym odcinku czasu i rywalizującymi o te zasoby, uwzględniający wymagania SLA, czy bardziej ogólnie wymagania jakościowe QoS (*Quality of Service*),
- dostarczanie (przypisanie) (*provisioning*) odpowiedniej wielkości zasobów dla konkretnego żądania jako pewnej części zasobów przydzielonych żądającemu użytkownikowi w jego umowie z dostawcą,
- szeregowanie (*scheduling*) zasobów, kiedy zadany jest już zbiór zadań do wykonania oraz znana jest wielkość dostarczonych zasobów i ustala się porządek (start i zakończenie) wykonania poszczególnych zadań na tych zasobach oraz ewentualne uwzględnienie innych dodatkowych uwarunkowań (np. relacji między zadaniami).

Strategia zarządzania powinna uwzględniać też bieżącą dynamikę zmian żądań użytkowników. Wówczas istotny jest związek (*correspondence)* pomiędzy zasobami wymaganymi do wykonania danej usługi (inaczej odpowiadającego jej zadania), a zasobami obliczeniowymi możliwymi do wykorzystania w danym momencie, oferowanymi przez chmurę. Na ogół wykorzystuje się odpowiednie narzędzia, z jednej strony wspomagające predykcję zasobów niezbędnych do wykonania konkretnych usług, z drugiej zaś zdolnych do określenia listy wolnych zasobów możliwych do zaangażowania w celu wykonania tych żądań, przy uwzględnieniu konkretnej infrastruktury obliczeniowej chmury. W tym drugim przypadku istotną rolę odgrywają odpowiednie systemy monitorowania zasobów.

W praktyce chodzi o zapewnienie jak największej elastyczności funkcjonowania chmury przy minimalnych kosztach jej eksploatacji, jak też największej wydajności. Tak zdefiniowany problem optymalizacyjny dotyczący chmur obliczeniowych powinien być rozwiązywany i realizowany przez odpowiednie pakiety zarządzania zadaniami [1], dostępne w chmurze, które spełniają zadane kryteria optymalizacji wykorzystania zasobów, w tym na przykład prowadzą do równoważenia obciążeń serwerów chmury.

Mogą być rozpatrywane różne kryteria optymalizacji, dotyczące zasobów fizycznych chmury (stopień wykorzystania procesora czy pamięci) lub oferowanych

usług (liczba czy długość kolejki oczekujących usług, czy czas odpowiedzi na żądanie użytkownika przy zadanej przepustowości łączy). Uwzględniane są różne dodatkowe parametry jak wielkości graniczne dostępnych zasobów czy przeciążenia lub niedociążenia wskazanych zasobów w trakcie eksploatacji chmury obliczeniowej. Na ogół uwzględnia się kilka różnych kryteriów jednocześnie. Wielokryterialne problemy zarzadzania zasobami chmurowymi nie są problemami o wielomianowej złożoności obliczeniowej, gdyż liczba wszystkich możliwych wariantów dostarczania zasobów chmurowych dla konkretnego zestawu żądań użytkowników przy założonym okresie obserwacji, jest ogromna i może być określana co najmniej funkcją wykładniczą [10]. Dlatego rozpatruje się wiele algorytmów heurystycznych zarządzania zasobami chmury, jedne z nich dotyczą wykorzystania algorytmów genetycznych inne zaś uczenia maszynowego [17].

W przypadku algorytmów genetycznych popularne są algorytmy mrowiska czy ławicy ryb. Pierwsze z nich wykorzystują metodę optymizacji kolonii mrówek (mrowiska) (*ant colony optimization* - ACO), która w sposób dynamiczny przypisuje zasoby chmury do zmieniających się wymagań użytkowników, jednocześnie zapewniając powolną zbieżność do ustalonego rozwiązania. Szacuje się wskaźnik zainteresowania zasobami chmury (wagi, priorytety) oraz dokonuje się ich przypisania do usług zarówno na podstawie wyróżnionych cech jak i zadanych z góry kryteriów QoS (czas wykonania, wydajność, bezpieczeństwo czy koszt) [13]. Drugi typ algorytmów związany jest z wykorzystaniem metaheurystyk w optymalizacji wielokryterialnej dotyczących przypisania zadań do zasobów uwzględniających reguły zachowania się ławicy ryb (np. przemieszczania się obok siebie, trzymania się razem czy unikania kolizji) i charakteryzujących się szybszą zbieżnością i dokładnością rozwiązania, ale przy założeniu wykorzystania zadanej wersji wyjściowej [15]. W przypadku uczenia maszynowego wykorzystuje się w tym celu różne modele statystyczne (ARMA, ARIMA czy ARMAX) w celu predykcji określonych metryk na podstawie historycznej obserwacji funkcjonowania chmury [7]. Jedną z wad takich metod jest trudność zaprojektowania efektywnej topologii sieci neuronowej i powolność jej uczenia. Lepsze wyniki (częściej znajdowanie globalnego a nie lokalnego rozwiązania przy racjonalnym czasie uczenia sieci) zapewnia podejście SVM (Support Vector Machine) dla predykcji wielkości wymagań użytkowników na podstawie danych historycznych oraz wykorzystanie teorii kolejek (model M/M/c) do określenia optymalnej liczby zasobów dla zrealizowania tych żądań przy redukcji czasu realizacji usług oraz uniknięcia nadmiernego obciążenia wykorzystanej infrastruktury [17].

W tym artykule zaproponowano uogólniony model zarządzania usługami, przeanalizowano optymalizację procesów zarządzania dotyczących dostarczania zasobów dla żądań użytkowników współczesnych rozwiązań chmurowych. Podano uogólniony model zarządzania oraz wskazano heurystyczne algorytmy alokacji wykorzystujące metody sztucznej inteligencji. Szczególną uwagę zwrócono na uproszczenie metody zarządzania oraz na potrzebę uczenia maszynowego w trakcie funkcjonowania chmury. Przedstawiono również przykład konkretnego rozwiązania dotyczącego rozwijanej własnej chmury TASKcloud [9], potwierdzający zasadność proponowanego podejścia.

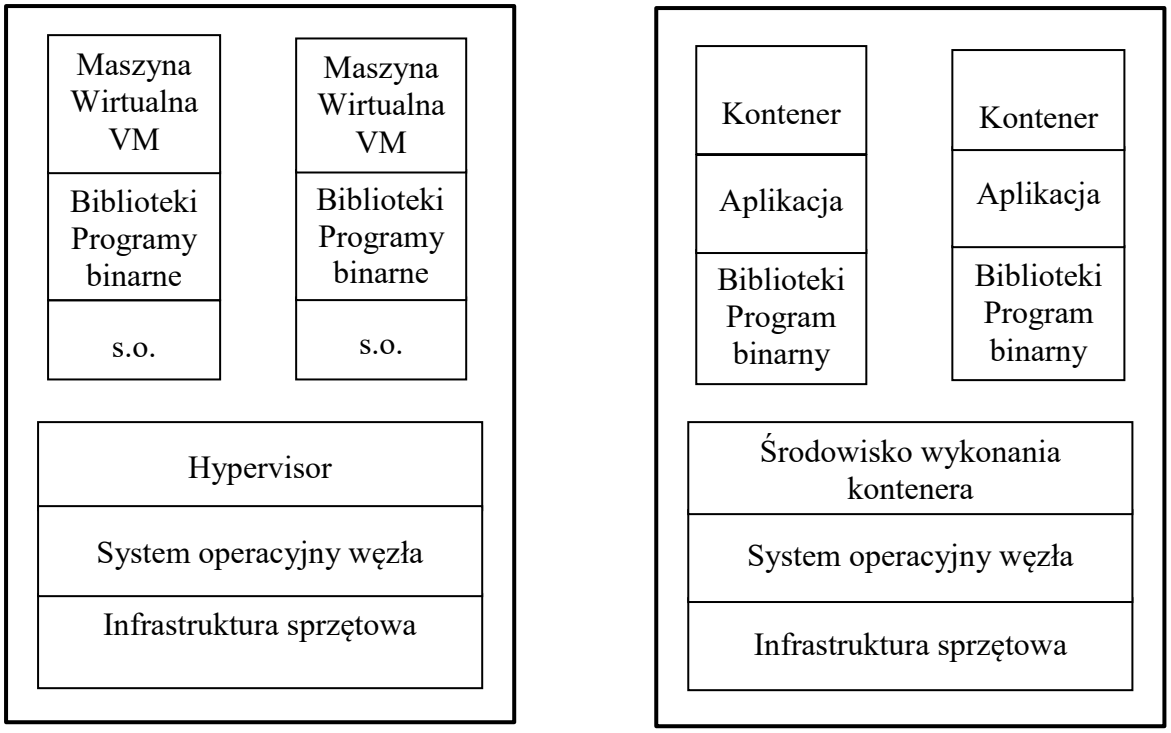
2. Uogólniony model zarządzania zasobami chmurowymi

Jak wspomniano we wstępie, obserwowany trend metod przetwarzania usług wiąże się z integracją różnych rozwiązań dotyczących Internetu Rzeczy (IoT), przetwarzania w chmurze oraz dużych zbiorów danych. IoT dotyczy wykorzystania różnego typu urządzeń cyfrowych, które dostarczają dużej ilości danych, umożliwiając zdalne monitorowanie i zarządzanie obiektami na których się te urządzenia znajdują. Z kolei lokowanie chmury obliczeniowej blisko takich źródeł danych (*big data*) redukuje obawy związane z opóźnieniem w transmisji danych. Poza tym rozwiązania typu *Serverless* idą jeszcze dalej, gdyż aplikacje zaprojektowane i przygotowane do pracy bez serwera (maszyny fizycznej czy też wirtualnej) mogą być łatwo modyfikowane wraz z możliwością uruchomienia w różnych lokalizacjach dostawcy chmury. Istotne staje się też zarządzanie danymi w celu zapewnienia dostępności i efektywnego ich wykorzystania. Jak dotąd ekonomiczne wykorzystanie zasobów chmury jest ciągle istotnym problemem, mówi się nawet o tzw. marnotrawstwie zasobów chmury. Ten fakt ogranicza w dużym stopniu wprowadzaniu chmury do wielu organizacji. Dlatego też dostawcy chmur obliczeniowych koncentrują się na opracowywaniu nowych efektywnych narzędzi i platform do zarządzania i przetwarzania w chmurze, chcąc poprawić aspekty ekonomiczne jej eksploatacji. Użytkownicy chmury są również zainteresowani zarówno konkurencyjnymi kosztami, jak też zapewnieniem wysokiej wydajności chmury dla swoich usług. Na ogół muszą oni zarządzać wieloma usługami jednocześnie, uwzględniając potrzeby różnych kontrahentów, związanych z wieloma dostawcami chmury. Stąd obserwuje się potrzebę wielokryterialnej optymalizacji oraz wykorzystania procedur automatycznego zarządzania zasobami. W konsekwencji istnieje potrzeba opracowania coraz efektywniejszych metod i algorytmów przypisania żądań użytkowników końcowych (usługobiorców) chmury do jej zasobów.

W dalszych rozważaniach uwzględnimy dwie architektury chmurowe oparte o maszyny wirtualne (*Virtual Machine* - VM) oraz kontenery (*containers*), których schematy przedstawiono na rysunku 1. Dzięki wirtualizacji jest możliwe tworzenie wielu maszyn wirtualnych na jednej maszynie fizycznej. Każda maszyna wirtualna posiada własny system operacyjny (OS) i aplikacje, które są przechowywane jako tzw. obrazy. Maszyna wirtualna potrzebuje lekkiej warstwy oprogramowania określanej jako *hypervisor*, która koordynuje komunikację między maszyną a infrastrukturą sprzętową. Ta warstwa przydziela fizyczne zasoby obliczeniowe: procesory, pamięć operacyjną oraz pamięć masową i urządzenia sieciowe, poszczególnym maszynom wirtualnym. Dzięki temu każda z nich działa niezależnie, nie zakłócając pracy pozostałych. Co więcej takie rozwiązanie umożliwia uruchamianie różnych systemów operacyjnych (w skrócie s.o.) na tym samym serwerze, tym samym zapewnia bardziej wydajne i ekonomiczne wykorzystanie zasobów fizycznych.

Rozwiązania kontenerowe opakowują kod i wszystkie zależne biblioteki, pliki binarne czy konfiguracyjne, tworząc też tzw. obraz. Zamiast wirtualizować całość systemu operacyjnego kontenery opakowują w uniwersalny sposób aplikację z niezbędnymi pakietami dzięki czemu są niezależne od oprogramowania obecnego na systemie gościa (hosta). Jednocześnie są przenośne i dają możliwość testowania takiej niezmiennej jednostki. Kontenery korzystają z systemu operacyjnego gościa, dlatego nie trzeba na nich uruchamiać systemu operacyjnego ani ładować jego bibliotek. To

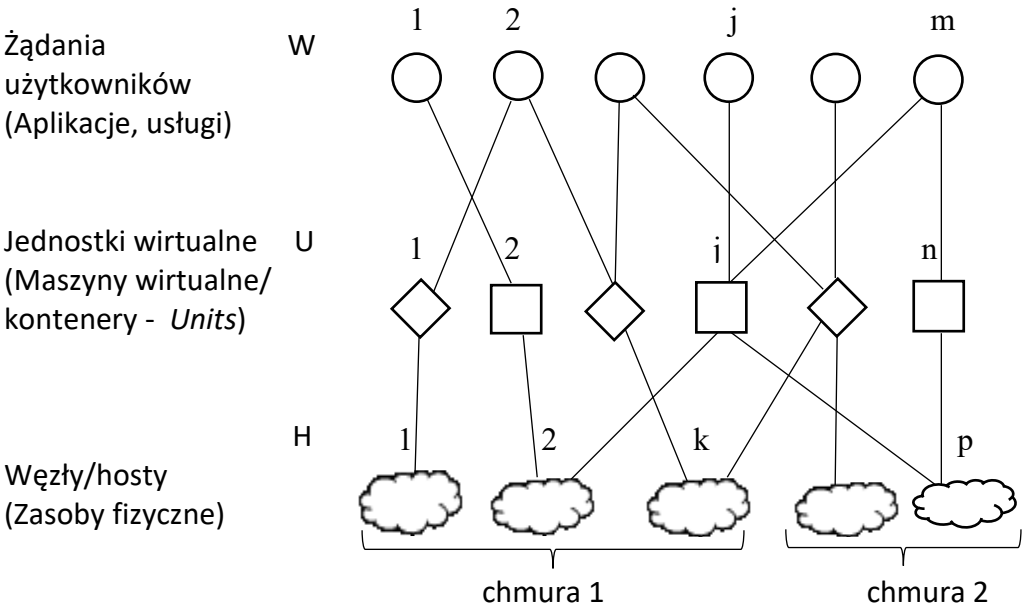
sprawia, że kontenery są mniejsze (zajmują mniej miejsca na dysku), a tym samym szybkie i przenośne. Kontener jest uruchomioną instancją obrazu, ale z jednego obrazu możemy uruchomić wiele instancji – kontenerów zmieniając im wyłącznie konfigurację, dzięki czemu otrzymujemy wiele instancji działających jednocześnie. Obraz kontenera to zatem prosty, samodzielny, wykonywalny pakiet oprogramowania (zadanie - *workload*), który zawiera wszystkie komponenty niezbędne do uruchomienia aplikacji: kod źródłowy, środowisko uruchomieniowe (*runtime*), narzędzia systemowe, biblioteki systemowe i pliki konfiguracyjne (przy czym część z nich możliwa jest do zamontowania dopiero przy uruchamianiu kontenera)



a) b)

Rys. 1. Architektury podstawowych jednostek wirtualnych: a) maszyna wirtualna b) kontener

Na rysunku 2 zaproponowano uogólniony model realizacji żądań użytkowników mających na celu wykonanie odpowiednich usług czy aplikacji na zasobach fizycznych chmury. Zakładamy, że każde wykonanie instancji usług czy aplikacji na maszynach wirtualnych czy w kontenerach angażuje odpowiednie (z reguły nie wszystkie) zasoby chmury przewidziane w kontrakcie SLA. Nie odbywa się to jednak w sposób bezpośredni. Przedstawiony model złożony jest z trzech warstw, gdzie wyszczególnione warstwy odpowiadają kolejno: żądaniom użytkowników, jednostkom wirtualnym (maszynom wirtualnym lub kontenerom), węzłom obliczeniowym (hostom) chmury lub chmur. Każdy węzeł (*host*) reprezentuje sobą określone zasoby fizyczne, nie pokazane na rysunku 2, przedstawiające sobą konkretne możliwości obliczeniowe, które powinny być dopasowywane do potrzeb oczekujących na wykonanie usług i aplikacji.



Rys. 2. Koncepcja dwustopniowego przyporządkowania usług i aplikacji jednostkom wirtualnym oraz jednostek wirtualnych węzłom (hostom) chmur obliczeniowych

Bazując na modelu przedstawionym na rysunku 2 oraz znając wielkości parametrów zasobów fizycznych można oszacować maksymalną liczbę instancji dla danej usługi czy aplikacji na danym serwerze. Niestety ta liczba zmienia się dynamicznie z uwagi na dynamikę żądań użytkowników czy obciążenia węzłów. Poza tym heterogeniczność węzłów utrudnia wykonanie procedur optymalizacyjnych. Często też użytkownicy końcowi z uwagi na realizację złożonych scenariuszy usługowych mogą korzystać jednocześnie z dwu lub więcej chmur obliczeniowych. Takie rozwiązanie jest wymuszane ograniczoną dostępnością do unikatowych aplikacji bądź koniecznością zapewnienia wyższej wiarygodności przetwarzania (powielanie obliczeń). Wówczas istotne staje się uwzględnienie w procedurach optymalizacyjnych zwiększonych obciążeń komunikacyjnych. Poza tym użytkownicy mogą wskazywać na różne kryteria optymalizacji, które to mogą być często przeciwstawne. Na przykład zapewnienie minimalnego czasu wykonania scenariusza usług przy minimalnej liczbie zaangażowanych węzłów, albo minimalizacja konsumpcji energii chmury przy zachowaniu wysokiej wiarygodności działania.

Rozpatrzmy problem optymalizacji w sposób formalny. Załóżmy, że rejestrujemy żądania użytkowników w pewnym krótkim przedziale czasu. Oznaczmy przez W zbiór zadań (*workloads*) związany z takimi żądaniami użytkowników, tzn. $W = \{w_1, w_1, ...w_i, ...w_{m-1}, w_m\}$, dostępny zbiór maszyn wirtualnych i kontenerów jako zbiór wirtualnych jednostek (*virtual units* - U) przez $U = \{u_1, u_1, ...u_j, ...u_{n-1}, u_n\}$, zaś zbiór węzłów (*hosts*) chmury przez H, gdzie $H = \{h_1, h_1, ...h_k, ...h_{p-1}, h_p\}$. Wprowadźmy dwa odwzorowania X oraz Y, których elementy x_{ij}, oraz y_{jk} określają przypisanie zadań do wirtualnych jednostek oraz wirtualnych jednostek do węzłów.

Przyjmijmy, że $X = [x_{ij}]$, gdzie:

$$x_{ij} = \begin{cases} 1 - jeśli\ w_i\ \ jest\ \ przypisane\ do\ u_j \\ 0 - w\ \ przeciwnym\ \ przypadku \end{cases} \tag{1}$$

$i = 1,2,.....,m \quad j = 1,2,.....,n \quad m \geq n$

Analogicznie $Y = [y_{jk}]$, gdzie:

$$y_{jk} = \begin{cases} 1 - jeśli\ u_j\ \ jest\ lokowane\ na\ h_k \\ 0 - w\ \ przeciwnym\ \ przypadku \end{cases} \tag{2}$$

$j = 1,2,.....,n \quad k = 1,2,.....p \quad n \geq p$

Zauważmy, że jeśli $\sum_{i=1}^{m} x_{ij} = 1$ to oznacza to, że każde w_i jest przypisane do jednego u_j, co nie wyklucza, że u_j może zawierać kilka w_i. Podobną uwagę można odnieść do przyporządkowania y_{ik}.

Tak więc macierze X i Y określają konkretne przypisanie zasobów do zadanych żądań użytkowników, co oznaczamy przez *P(X, Y)*. Zakładając pełne wykorzystanie dostępnej infrastruktury chmurowej, liczba możliwych przyporządkowań zadań ze zbioru W do jednostek wirtualnych chmury ze zbioru U wynosi Ω_1, gdzie:

$$\Omega_1 = \binom{m}{n} \cdot n! = \frac{m!}{(m-n!)} \tag{3}$$

W podobny sposób można oszacować liczbę możliwych przyporządkowań jednostek wirtualnych (U) do węzłów chmury (H), wówczas:

$$\Omega_2 = \binom{n}{p} \cdot p! = \frac{n!}{(n-p)!} \tag{4}$$

Zatem liczba wszystkich możliwych przypadków przyporządkowań dotycząca modelu dwustopniowego wyniesie Ω, gdzie:

$$\Omega = \Omega_1 \cdot \Omega_2 = \frac{m! \cdot n!}{(m-n)!(n-p)!} \tag{5}$$

Na przykład dla $m = 500, n = 100, p = 50$, oszacowanie odgórne liczby wszystkich wariantów wyniesie:

$$\Omega = \frac{500! \cdot 100!}{400! \cdot 50!} = 401 \cdot 402 \cdots 499 \cdot 500 \cdot 51 \cdot 52 \cdots 99 \cdot 100 << 500 \cdot 100 \cdot 100 \cdot 50 = 25 \cdot 10^7 \tag{6}$$

Tak ogromną liczbę przypadków trudno przeanalizować nawet wykorzystując komputery dużej mocy obliczeniowej. Konieczne jest uwzględnianie różnego typu ograniczeń dotyczących obrazów aplikacji czy usług lokowanych bezpośrednio do wskazanych maszyn wirtualnych, czy kontenerów. Istotne jest także wykorzystanie algorytmów heurystycznych zarzadzania, które znajdują przybliżone rozwiązania jak najbardziej zbliżone do rozwiązania optymalnego.

3. Parametry modelu i kryteria optymalizacji

W celu uszczegółowienia modelu zaproponowanego na rysunku 2, w tabeli 1 podano wykorzystane pojęcia oraz parametry je opisujące. W zależności od przyjętej

metody zarządzania podane parametry mogą być uszczegóławiane oraz opracowywane konkretne metody ich pomiaru. Poza tym mogą one stanowić podstawę formułowania odpowiednich kryteriów optymalizacji procedur zarządzania zadaniami i zasobami.

Tabela 1

Podstawowe wielkości oraz parametry modelu zarządzania aplikacjami i usługami

Nazwa	Opis	Parametry
Żądania użytkowników	Wykonanie pojedynczej usługi lub aplikacji jako scenariusza, powiązanych usług (graf)	Liczba użytkowników, częstość pojawiających się żądań, kategorie żądań, wymagany czas obsługi
Zadania do wykonania	Obudowany program wykonywalny (kod, biblioteki, dane) w postaci zadania przygotowanego do wykonania na infrastrukturze chmurowej	Zbiór zadań do wykonania w zadanym przedziale czasu, charakterystyka każdego z zadań, czas wykonania pojedynczego zadania, czas transmisji danych
Wirtualne jednostki chmury	Maszyna wirtualna lub kontener instalowany na infrastrukturze (węzłach) chmury	Liczba dostępnych maszyn wirtualnych, liczba dostępnych kontenerów, bieżący stopień zaangażowania węzłów chmury
Fizyczne zasoby chmury	Określone możliwości obliczeniowe węzłów chmury	Szybkość działania procesora, wielkość pamięci, przepustowość sieci, wielkość pamięci dyskowej
Środowisko chmurowe	Pojedyncza chmura obliczeniowa bądź zestaw chmur współpracujących dzięki ujednoliconym interfejsom zewnętrznym	Liczba chmur współpracujących, poziom współpracy, architektura poszczególnych chmur, poziom dostępności usług
Zarządzanie usługami	Narzędzia wspomagające organizację procesu zarządzania i monitorowanie na poziomie wielu chmur, pojedynczej chmury oraz jej poszczególnych węzłów	Czas obsługi żądania, czas szacowania wymaganych zasobów, czas przypisania usług do zasobów, czas wykonania procedury zarządzania

Analizując dane zawarte w tabeli 1 można sformułować kilka typowych kryteriów optymalizacji:

- Minimalizacja czasu realizacji usług ze zbioru U zgodnie z przyporządkowaniem $P(X,Y)$:

Czas realizacji pojedynczej usługi może być określony poprzez trzy składowe: czas ulokowania odpowiedniego zadania na zasobach wirtualnych chmury obliczeniowej, czas transmisji danych do tej jednostki wirtualnej oraz czas wykonania tego zadania na wskazanych zasobach fizycznych węzła chmury.

- Minimalizacja konsumpcji energii dla wykonania usług zgodnie z przyporządkowaniem $P(X,Y)$:

W tym przypadku dla konkretnej usługi istnieje konieczność określenia wielkości zużytej energii w powyższych trzech przypadkach, co oznacza pomnożenie wyżej wymienionych czasów: obliczeń i transmisji danych poprzez odpowiednie współczynniki zużycia energii.

- Minimalizacja intensywności uszkodzeń w chmurze obliczeniowej:

Przy tego typu kryterium trzeba rozpatrzyć zarówno intensywność błędów pojawiających się w jednostce wirtualnej jak też intensywność uszkodzeń węzła oraz zapewnić zwielokrotnienie wykonania tych samych usług.

- Minimalizacja czasu migracji jednostek wirtualnych na inne zasoby fizyczne chmury dla przyporządkowania *P(X,Y)*:

Jest to konieczne w przypadku nadmiernego obciążeniem węzła chmury bądź jego uszkodzenia. Czas ten obejmuje czas uruchomienia nowego węzła, jeśli nie jest aktywny oraz czas transmisji jednostek wirtualnych pomiędzy dwoma węzłami: rozpatrywanym i tym docelowym, a także czas wyboru i posadowienia wskazanej usługi na węźle docelowym.

Oznaczmy przez *T(X,Y)* czas wykonania usług ze zbioru U dla przyporządkowania *P(X,Y),* zaś Z_k wielkość zasobów k-tego węzła wykorzystywanych po tym przyporządkowaniu. Wówczas kryterium optymalizacji dla zapewnienia zrównoważenia obciążenia i maksymalizacji liczby użytkowników w zadanym przedziale czasu można sformułować następująco:

$$\text{Znaleźć } P(X,Y) \text{ takie, że } T(X,Y) = \text{minimum}$$

$$\text{oraz} \quad \sum_{i=1}^{m} x_{ij} = 1;\ \sum_{j=1}^{p} y_{ij} = 1;\ Z_k \le Z_o \text{ dla } \forall k \in \{1,2,...,p\} \tag{6}$$

Z_o – wykorzystywane wielkości zasobów, jednakowe dla wszystkich węzłów, w rozdziale następnym wyjaśniono jakie elementy zawierają zbiory Z_k oraz Z_o.

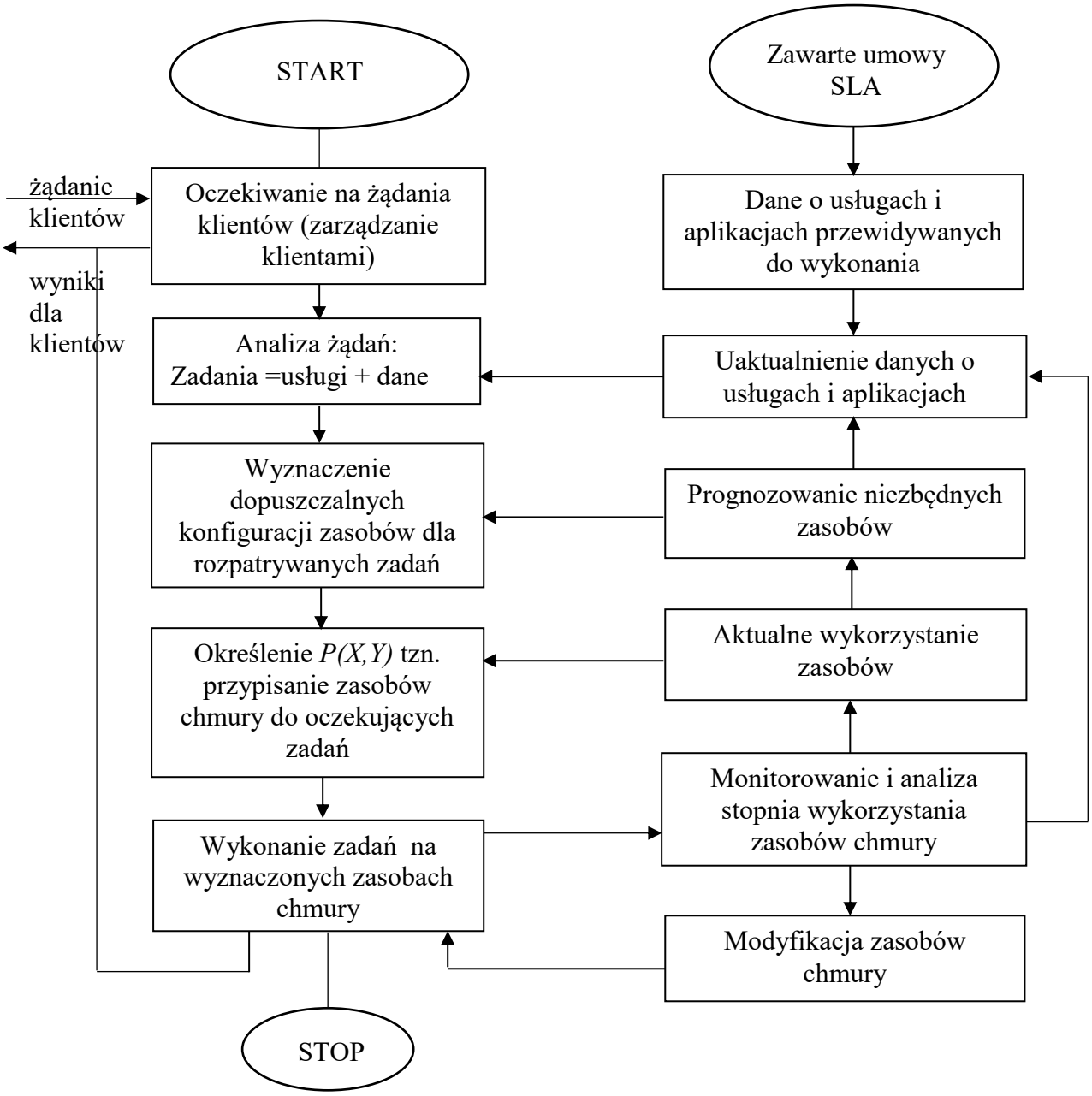
Każde z wyżej wymienionych kryteriów wymaga więc pozyskania dodatkowych danych oceniających różne warianty wyboru zasobów węzła. Dane te mogą być dostarczone dzięki dodatkowym eksperymentom wykonywanych na badanej chmurze obliczeniowej, albo estymowane na podstawie dostępnych symulatorów modelujących zachowanie się takiej chmury obliczeniowej. Tego typu dane mogą być też uzyskane poprzez budowę i analizowanie odpowiednich modeli teoretycznych opisujących zachowanie się badanej chmury. W każdym z tych przypadków pojawia się problem kompletności i adekwatności tak uzyskanych danych, a w konsekwencji jakości opracowanych na tej podstawie procedur zarządzających.

4. Strategia zarządzania usługami

Konkretna strategia zarządzania usługami chmury polega na organizowaniu zasobów dla wykonania pojedynczej usługi lub scenariusza zależnych lub niezależnych usług,

spełniających przyjęte kryteria optymalizacji. Można założyć, że mamy do zrealizowania pewien ciąg działań, poczynając od rejestracji pojawiającego się żądania użytkownika poprzez ocenę wymagań niezbędnych do zrealizowania odpowiadającego mu zadania oraz przydziału wymaganych zasobów i wykonaniu na nich tego zadania. Jeśli nie jest utworzony w jednostce wirtualnej obraz usługi odpowiadający takiemu zadaniu, to dodatkowo należy to uczynić, by następnie wskazać węzeł chmury oraz odpowiednie jego zasoby do wykonania tego zadania. Wybór węzła nie jest dowolny, a powinien on spełniać w jak najlepszym stopniu przyjęte kryteria optymalizacyjne. Zgodnie z modelem z rys. 2, wybrano postępowanie z dwóch stron; po pierwsze: analizując z góry żądania użytkowników oraz po drugie: określając od dołu dostępne zasoby węzłów obliczeniowych. Następnie jednostkom wirtualnym z jednej strony nadajemy parametry odpowiadające poszczególnych żądaniom użytkowników, z drugiej zaś przypisujemy dostępne do wykorzystania zasoby węzłów obliczeniowych. Na rysunku 3 przedstawiono schemat zarządzania usługami uwzgledniający takie podejście. Wyróżniamy w nim cztery elementy: prognozowanie niezbędnych zasobów dla realizacji żądania, pozyskiwanie wiedzy o aktualnym wykorzystaniu zasobów chmury oraz uruchomienie algorytmu wyboru najodpowiedniejszego węzła i jego zasobów dla wykonania zadania odpowiadającego temu żądaniu, a także jego wykonanie. Aktualne wykorzystanie zasobów chmury można pozyskać z systemu monitorowania węzłów chmury. Budowa takiego systemu monitorującego może być również dużym wyzwaniem, ale przyjęto, że takie procedury pomiarowe istnieją, są wykonywane na poziomie węzłów chmury i mogą dotyczyć bieżącego obciążenia nawet poszczególnych jego procesorów czy rdzeni.

Rozpatrzmy zadanie w_i wykonywane przez węzeł h_k na zasobach rzeczywistych Z_k, gdzie zasoby te przedstawiają parametry: $z_{rk} \in Z_k$, $r = 1,2,3,4$, które opisują odpowiednio: szybkość procesora w MIPS/FLOPS (z_{1k}), wielkość pamięci MB (z_{2k}), obszar dysku w GB (z_{3k}) oraz przepustowość sieci w MB/s (z_{4k}). Z kolei wartości estymowane tych parametrów oznaczone są przez Z_k^* (z^*_{rk}). Oczywiste jest, że na ogół wartości estymowane (z^*_{rk}) różnią się od wartości rzeczywiście wykorzystywanych (z_{rk}). Warto też podkreślić, że wykonanie i-tej usługi na k-tym węźle jest możliwa wtedy gdy wielkość wymaganych zasobów jest nie większa niż wielkość dostępnych zasobów na tym węźle. Optymalne przypisanie $P(X,Y)$ w dużej mierze zależy od dokładności szacowania niezbędnych zasobów, co jest głównym problemem rozpatrywanym w tym artykule. Jeśli te wartości są zbyt wysokie, zarządca chmury przydziela węzłowi zbyt dużą liczbę zasobów z których część pozostanie niewykorzystana. Jeśli natomiast szacowanie jest zbyt niskie, usługa otrzyma zasoby niewystarczające do jej wykonania. Wówczas konieczna staje się migracja jednostki wirtualnej na inny węzeł, co z kolei jest czasochłonnym działaniem.



Rys. 3. Proponowana metoda zarządzania usługami chmury obliczeniowej zgodna z modelem z rys. 2

Warto jeszcze raz podkreślić, że specyfikacja żądań użytkowników nie musi bezpośrednio dotyczyć wymagań na zasoby sprzętowe dostępne przez węzły chmury. Może określać jedynie pewne cechy funkcjonalne, takie jak kategoria usługi czy parametry wysokopoziomowe (tabela 1). Zadaniem programu zarządzającego, a w szczególności modułu prognozowania niezbędnych zasobów (rys. 3) jest odpowiednie przełożenie tych specyfikacji na poziom wielkości zasobów. Do estymacji niezbędnych zasobów proponuje się zastosowanie metod sztucznej inteligencji i wykorzystanie odpowiednio wyuczonej sieci neuronowej o architekturze złożonej z jednej warstwy ukrytej. Co więcej model takiej sieci neuronowej może być tworzony dynamicznie, jeśli program zarządzający, a konkretnie baza zarejestrowanych danych

posiada wystarczający zestaw próbek do uczenia tej sieci w trakcie funkcjonowania chmury obliczeniowej. Takie dynamiczne tworzenie modelu sieci neuronowej może wydłużyć czas obsługi żądania użytkownika, ale z uwagi na niewielki rozmiar zastosowanej sieci neuronowych, czas zarzadzania siecią powinien być do zaakceptowania. Tak więc nieprzerwalne funkcjonowanie chmury w dłuższej perspektywie z reguły zapewnia wydobycie właściwych danych uczących i co więcej zapewnia kalibrację przyjętych sieci neuronowych równolegle z realizacją zadań podstawowych chmury. Zgodnie z wiedzą autorów tego artykułu, takie podejście nie było dotychczas analizowane i wykorzystywane w praktyce.

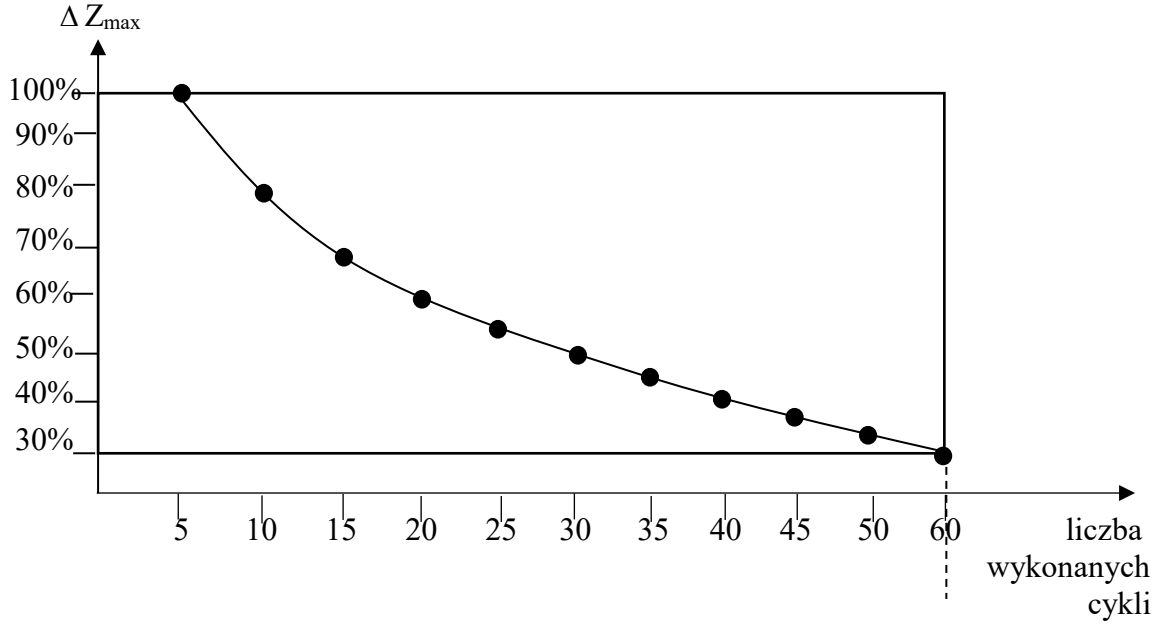
Do przetestowania proponowanej metody prognozowania wymaganych zasobów dla zgłaszanych żądań użytkowników wykorzystano chmurę obliczeniową TASKcloud [9], opracowaną i eksploatowaną w CI TASK od kilku lat. Cechą charakterystyczną jest wykorzystanie do jej budowy oprogramowania otwartego (np. OpenStack, Ceph, Hadoop), co czyni takie rozwiązanie uniwersalnym, a wyniki eksperymentów mogą być, w pewnym stopniu, odnoszone do innych rozwiązań chmurowych wykorzystujących tego typu oprogramowanie. Założono również odpowiednią politykę przypisania usług do węzłów, tak by zapewnić:

1. Zrównoważenie obciążenia we wszystkich aktywnych węzłach obliczeniowych,
2. Minimalizację czasu wykonania usługi,
3. Maksymalizacja liczby użytkowników korzystających z usług.

Tego typu kryterium optymalizacyjne podaje wzór (6). W badaniach rozpatrzono kilka kategorii usług zgłaszanych przez użytkowników, takich jak usługi internetowe czy mobilne, usługi gromadzenia i przetwarzania danych, usługi multimedialne, czy obliczenia naukowe. Przyjęto też kilka parametrów opisujących usługi jak i zasoby, które zostały przedstawione w Tabeli 1.

Na ogół dobór parametrów wejściowych sieci neuronowej uzależniony jest od kategorii rozpatrywanych usług. W przyjętych rozważaniach skupiono się na usługach typu obliczenia naukowe wymagających dużej mocy obliczeniowej z uwzględnieniem reprezentatywnych wielkości danych. W takim przypadku istotną rolę odgrywa nie tylko rozmiar przetwarzanych danych ale też liczba wykonywanych instrukcji. Parametry wyjściowe sieci neuronowej stanowią estymowane wielkości poszczególnych zasobów sprzętowych niezbędne do wykonania zadania o parametrach podanych na wejściu takiej sieci.

Dane uczące sieć można wyznaczyć eksperymentalnie poprzez wykonanie na węzłach chmury odpowiednich programów wzorcowych (*benchmarks*) przy jednoczesnym monitorowaniu wielkości wykorzystanych zasobów. Na tej podstawie wyuczona sieć neuronowa może prognozować wielkość niezbędnych zasobów dla danej kategorii usług. Takie podejście zastosowano w przyjętym schemacie zarządzania (rys. 3) wykorzystując do pozyskania danych uczących aplikacje wzorcowe: HPCG (*High Performance Conjugate Gradients*), stosowany do oceny szybkości działania superkomputerów przy kwalifikacji na listę Top 500 oraz HPCC (*High Performance Computing Cluster)* wykorzystywany przy analizie ogromnych zbiorów danych (*Big Data*). Po wstępnym nauczeniu sieci neuronowej takimi danymi, była ona następnie douczana danymi gromadzonymi podczas monitorowania rzeczywiście wykonywanych zadań zlecanych przez różnych użytkowników.



Rys.4. Maksymalna różnica w ocenie niezbędnych zasobów estymowanych i rzeczywiście wykorzystywanych dla zaakceptowanego przypisania $P(X,Y)$

Istotna jest ocena precyzji estymacji zasobów do wykonania zgłaszanych usług. Z uwagi na zmienność żądań użytkowników oraz dynamikę wykonywanych obliczeń w chmurze nie zastosowano miar statystycznych. Ograniczono się jedynie do wyliczania maksymalnych różnic procentowych dotyczących wartości estymowanych oraz mierzonych (rzeczywistych) wielkości wykorzystywanych zasobów. Za miarę oceny estymacji zasobów przyjęto następującą definicję:

$$\Delta Z_{\max} = \max_{\substack{Z_{rk} \in Z_k \\ k=1,2,..,p}} \left(\frac{\left| z_{rk} - z^{*}_{rk} \right|}{z_{rk}} \cdot 100\% \right) \tag{7}$$

Na rysunku 4 przedstawiono tendencję zmian maksymalnej wielkości $\Delta Z_{\max}$ przy zwiększaniu liczby cykli uczenia zastosowanej sieci neuronowej. Z przeprowadzonych eksperymentów wynika, że poprawność estymacji zasobów na początku wykorzystania sieci neuronowej była znacznie mniejsza, ($\Delta Z_{\max} = 100\%$), zaś po 60 cyklach wykonania procedury zarzadzania i równoległego uczenia sieci neuronowej zwiększyła się ponad trzykrotnie ($\Delta Z_{\max} = 30\%$). Co więcej w tym ostatnim przypadku dla 80% dokonanych estymacji maksymalny rozrzut $\Delta Z_{\max}$ wynosił już tylko 20%. Z badań eksperymentalnych przeprowadzonych dla innych kategorii zgłaszanych usług otrzymujemy podobną tendencję zmian $\Delta Z_{\max}$.

4. Wnioski końcowe

W pracy wykazano, że zarzadzanie usługami chmury obliczeniowej jest zadaniem złożonym i wymaga podjęcia się wielu czynności pomocniczych dotyczących między innymi szacowania wielkości zasobów niezbędnych do obsługi żądań użytkowników.

Niektóre z nich można określić na podstawie wymagań zawartych w porozumieniu SLA. Dotyczą one raczej zasad wykorzystania chmury obliczeniowej przez danego użytkownika oraz zobowiązań dostawcy chmur obliczeniowych bądź dostawcy usług. Zawarte tam informacje należy więc odnieść do parametrów niezbędnych do zarzadzania usługami chmury oraz uzupełnić je o dodatkowe dane które umożliwią zaimplementowanie efektywnych procedur zarządzających. W artykule przedstawiono dwustopniowy model zarzadzania chmurami, scharakteryzowano podstawowe parametry tego modelu oraz podano pewne wielowymiarowe kryteria optymalizacyjne. Na tej podstawie opracowano odpowiedni schemat zarządzania chmurą. Następnie rozpatrzono problem szacowania niezbędnych zasobów chmury do wykonania zadanej kategorii usług. W tym celu zaproponowano wykorzystanie dynamicznie uczącej się sieci neuronowej wskazującej właściwy zestaw zasobów fizycznych dla żądań opisanych konkretnymi wartościami parametrów. Eksperymentalnie wykazano że proponowana metoda jest praktycznie akceptowalna.

Kompleksowa strategia zarządzania powinna jednak integrować działania na wszystkich poziomach modelu zaproponowanego na rys. 2. Wymaga to dostępności do wielu danych, które powinny być gromadzone podczas funkcjonowania chmur obliczeniowych. Dane te różnią się w zależności od konfiguracji chmury (pojedyncza, złożona z wielu chmur) czy wykorzystywanych różnych zasobów (np. IoT). Na ogół tego typu dane są własnością dostawców chmur czy usług i trudno je pozyskać dla badań naukowych. Dlatego w rozważaniach bazowano na eksperymentach wykonywanych na lokalnej chmurze obliczeniowej TASKcloud. Zauważono przy tym następujący paradoks, bardziej precyzyjne algorytmy zarzadzania wymagają znajomości coraz większej ilości precyzyjnych danych, co jest często trudno realizowalne w praktyce i nie musi prowadzić, z uwagi na dynamikę zmian zachodzących w chmurze obliczeniowej, do istotnego polepszenia opracowywanych rozwiązań. Dlatego też istotną alternatywą jest ograniczanie się do mniejszej ilości danych sterujących oraz do opracowania prostych, ale skutecznych rozwiązań, które mogą być automatycznie udoskonalane w trakcie funkcjonowania chmury obliczeniowej.

LITERATURA

1. Buyer's Guide. Cloud Management, 2022, https://www.peerspot.com/landing/report-cloud-management
2. Carvalho G., Cabral, B., Pereira, V. *et al*: Edge computing: current trends, research challenges and future directions. *Computing* **103,** 993–1023 (2021).
3. Dimitri N.: Pricing cloud IaaS computing services, Journal of Cloud Computing: Advances, Systems and Applications (2020) 9:14 https://doi.org/10.1186/s13677-020-00161-2
4. Gupta S., Agarwal I., Singh R.S.: A Novel Hybrid Algorithm for Workflow Scheduling in Cloud, International Journal of Cloud Computing, January 2021, DOI: 10.1504/IJCC.2023.10038837
5. Hong J., Dreibholz T., Schenkel J.A., Hu J.A.: An Overview of Multi-cloud Computing, Artificial Intelligence and Network Applications, 2019, Volume 927, pp 1055–1068

6. Islam S, Keung J, Lee K, Liu A.: Empirical prediction models for adaptive resource provisioning in the cloud. Future Generation Computer Systems 28(1):155–162 (2012),
7. James G. Witten D., Hastie T., Tibshirani R.: An Introduction to Statistical Learning with Applications in R, Springer 2016 (http://www-bcf.usc.edu/~gareth/ISL/index.html)
8. Kozhirbayev Z., Sinnott R.O.: A performance comparison of container-based technologies for the Cloud, Future Generation Computer Systems, Vol. 68, 2017, Pages 175-182, https://doi.org/10.1016/j.future.2016.08.025
9. Krawczyk H. (redaktor pracy zbiorowej): Efektywne wykorzystanie zasobów chmury obliczeniowej, wydawnictwo CI TASK, 2017, ISBN:978-83-947070-0-2
10. Li, C., Li, L.Y.: Optimal resource provisioning for cloud computing environment. *J Supercomputing* **62,** 989–1022 (2012). https://doi.org/10.1007/s11227-012-0775-9
11. Manvi S., Shyam G.K.: Cloud Computing Concepts and Technologies, CRC Press, Taylor & Francis Group, LLC, 2021
12. Masdari M., Nabavi S.S, Ahmadi V.,: An overview of virtual machine placement schemes in cloud computing, Journal of Network and Computer Applications,
13. Vol. 66, 2016, Pages 106-127, https://doi.org/10.1016/j.jnca.2016.01.011
14. Nigam S, Bajpai A.: An Optimal Resource Provisioning Algorithm for Cloud Computing Environment. Orient. J. Comp. Sci. and Technol;10(2). Available from: http://www.computerscijournal.org/?p=5573
15. Odun-Ayo I., Udemezue B., Kilanko A.: Cloud Service Level Agreements and Resource Management Advances in Science, Technology and Engineering Systems Journal Vol. 4, No. 2, 228-236 (2019), https://dx.doi.org/10.25046/aj040230
16. Ouyang M., Xi J., Bai W. and Li K.: Band-Area Application Container and Artificial Fish Swarm Algorithm for Multi-Objective Optimization in Internet-of-Things Cloud, in *IEEE Access*, vol. 10, pp. 16408-16423, 2022, doi: 10.1109/ACCESS.2022.3150326.
17. Sahi S.K., Dhaka J.V.S.: A Review on Workload Prediction of Cloud Services, International Journal of Computer Applications (0975 – 8887) Volume 109 – No. 9, January 2015
18. Vozmediano M., Montero R.S., Huedo E., Liorente I.M.: Efficient resource provisioning for elastic Cloud services based on machine learning techniques. Journal of Cloud Computing: Advances, Systems and Applications (2019) 8:5 https://doi.org/10.1186/s13677-019-0128-9