Raport projektu celowego
Obliczenia wielkiej skali i wizualizacja do zastosowań w wirtualnym laboratorium z użyciem klastra SGI

Zadanie WP 2.1. Zdalny dostęp do bibliotek naukowych

WP. 2.1.6. Mechanizmy szeregowania
oparte na wiedzy i analizie statystycznej
danych historycznych

Maciej Brzeźniak, Tomasz Makieła

Poznańskie Centrum Superkomputerowo-Sieciowe

Poznań, październik 2004 r.

Spis treści

1. Informacje ogólne

Niniejszy raport zawiera informacje dotyczące prac przeprowadzonych w ramach zadania WP. 2.1.6. projektu celowego p.n. „Mechanizmy szeregowania oparte na wiedzy i analizie statystycznej danych historycznych". Prace dotyczyły mechanizmów rozszerzających podsystem szeregowania zadań i planowania rozdziału zasobów dla systemu udostępniania bibliotek matematycznych w klastrze SGIgrid.

1.1. Cel prac

Stosowany w oryginalnym systemie NetSolve, (przyjętym jako baza dla usługi udostępniania bibliotek w SGIgrid) mechanizm szeregowania zadań posiada szereg ograniczeń.

Po pierwsze, istniejący mechanizm nie uwzględnia charakterystycznych cech danego środowiska obliczeniowego i jego dynamicznego charakteru (zmiany konfiguracji, polityk zarządzania itp.). Nie uwzględnia także heterogenicznej natury elementów środowiska, jest dostosowany do homogenicznych instalacji. Tymczasem klaster SGIgrid składa się z maszyn o różnej architekturze: Irix/SGI, Linux/PC itd., począwszy od stacji roboczych do superkomputerów. Charakterystyki tych maszyn są różne, m.in. odmienne są zależności pomiędzy stanem tych maszyn a wydajnością obliczeń, którą oferują.

Po drugie, architektura klastra SGIgrid narzuca stosowanie usług systemu Globus a także usług Brokera SGIgrid do dostępu do zasobów klastra. Co więcej, część maszyn nie jest dostępna dla usługi GRAM bezpośrednio, ponieważ jest zarządzana przez oddzielne systemy zarządzania zasobami takie jak LSF. Dla tych zasobów stosowane są specjalne job-manager'y rozszerzające funkcjonalność serwera GRAM. Przyjęta metoda dostępu do zasobów dodatkowo utrudnia szeregowanie. Standardowe, stosowane w NetSolve mechanizmy szeregowania nie sprawdzają się, ponieważ nie uwzględniają dodatkowych opóźnień i zasobów potrzebnych na uruchomienia zadania poprzez usługi GRAM i pozostałe systemy zarządzania zasobami takie jak LSF a także charakterystyki przetwarzania w systemach takich jak LSF.

Po trzecie, mechanizm szeregowania stosowany w NetSolve do oceny stanu zasobów obliczeniowych używa bardzo prostej metody oceny obciążenia systemu obliczeniowego i jakości połączenia sieciowego prowadzącego do węzła. Metoda oceny obciążenia węzła opiera się na tzw. *workload* (wyliczany na podstawie wskazań polecenia *uptime*, dokładniej *load average**100) na poszczególnych serwerach NetSolve. Dodatkowo Agent dodaje pewną wartość (100) do obciążenia danej maszyny przy przydzielaniu jej dla danego zadania obliczeniowego i odlicza tę wartość od obciążenia systemu po zakończeniu zadania. Ocena jakości połączenia sieciowego do węzła obliczeniowego obejmuje przepustowość i opóźnienie obserwowane na połączeniu Agent-Serwer NetSolve. Nie obejmuje jakości połączenia klient-serwer, która de facto decyduje o ilości czasu potrzebnego na przesłanie danych wejściowych dla obliczeń i odebranie wyników.

Po czwarte, mechanizmy szeregowania wykorzystywane w standardowym NetSolve nie czynią użytku z faktu, że wywołania funkcji bibliotek matematycznych w NetSolve wiążą się wykonywaniem tych samych, dobrze zdefiniowanych fragmentów kodu binarnego dla różnych danych wejściowych. Zdaniem PCSS analiza informacji dotyczących wydajności wielokrotnie powtarzanych wykonań kodu (na danym węźle obliczeniowym) w różnych warunkach (obciążenie systemu, łączy komunikacyjnych) pozwala na wyciąganie wniosków dotyczących przyszłej wydajności wykonania kodu na danym węźle (przy znanych wielkości danych wejściowych i wartości parametrów wykonania). Więcej informacji na ten temat można znaleźć w artykule opublikowanym przez PCSS na konferencji PPAM 2003 [PPAM2003].

Prace w ramach zadania WP.2.1.6 miały na celu rozszerzenie podsystemu szeregowania zadań tak by dostosować go do wymagań środowiska SGIgrid. Miały również zwiększyć dokładność przewidywania czasu wykonywania zadania na poszczególnych zasobach a przez to zwiększać sprawność mechanizmu szeregowania.

1.2. Zakres prac

Osiągnięcie celów określonych dla zadania wymagało następujących czynności:

a) definicja parametrów dotyczących wydajności wykonania funkcji matematycznej na zasobie:
   - parametry dotyczące wywołania funkcji matematycznej: rozmiar zadania, rozmiar danych wejściowych, przewidywany lub znany rozmiar danych wyjściowych, inne parametry wywołania (np. parametry modyfikujące działanie algorytmu)
   - informacje o zasobach zużytych przez wykonanie funkcji: CPU (system time, user time, iowait), liczba operacji i/o na sekundę, pamięć, użycie swap (objętość, ilość operacji) i inne
   - parametry dotyczące stanu systemu podczas wywołania funkcji: CPU load, total io, swap used/free, nr of swap operations, memory user/free etc.
   - parametry dotyczące jakości połączenia sieciowego: bandwidth, latency między agentem a serwerami
   - parametry dotyczące jakości połączenia sieciowego: bandwidth, latency między klientem a serwerami
   - parametry dotyczące repozytorium

b) opracowanie architektury systemu monitorującego:
   - metody pobierania niezbędnych informacji dotyczących wywołania funkcji: stanu systemu obliczeniowego podczas wywołania funkcji, zużytych zasobów (obliczenia, transmisja sieciowa, read/write wyników do repozytorium) itd.
   - metody ciągłego monitorowania stanu zasobów: cpu + związane, sieć, stan dysków repozytorium
   - metody zbierania danych dotyczących wydajności
c) opracowanie metod automatycznej analizy danych
   - badania wydajnościowych modeli przetwarzania maszyn o różnych architekturach a także modeli przetwarzania systemów obliczeniowych pracujących pod kontrolą zarządców zasobów,
   - badania mechanizmów odkrywania wydajnościowych charakterystyk wykonywania różnych klas zadań obliczeniowych na poszczególnych systemach komputerowych
d) opracowanie metod wizualizacji danych i wizualizacji wyników analizy (ważne dla weryfikacji mechanizmu przez administratora)
e) opracowanie metod szeregowania danych
   - zmiana mechanizmu oceny czasu wykonania
   - opracowanie mechanizmu oceny trafności predykcji
   - modyfikacja mechanizmu rozdzielania zasobów

W kolejnych punktach opisane zostaną prace wykonane w ramach zadania WP.2.1.6 oraz ich związek z pracami prowadzonymi w innych zadaniach projektu celowego.

2. Opis prac

W tym punkcie raportu zostały opisane prace wykonane w ramach zadania. Opis podzielony został na części analogicznie do podziału prac w punkcie 1.2.

2.1. Definicja parametrów dotyczących wydajności wykonania funkcji matematycznej na zasobie

Podstawowym źródłem danych dla systemu szeregowania zadań są dane o stanie systemów obliczeniowych oraz o przewidywanej ilości pracy koniecznej do wykonania danego zadania. W zależności od realizowanej polityki szeregowania, na podstawie znajomości tych czynników, przydziela się odpowiedni zasób, np. gwarantujący najkrótszy czas odpowiedzi lub zapewniający optymalne wykorzystanie zasobów środowiska.
Dlatego podstawowym zadaniem, które trzeba zrealizować dla poprawienia mechanizmu szeregowania dla środowiska SGIgrid jest rozszerzenie zakresu danych dotyczących zadań zlecanych przez użytkowników oraz rozszerzenie zakresu danych o stanie systemu analizowanych przez system predykcji i monitorowania. Muszą one uwzględniać specyficzne dla bibliotek matematycznych parametry zadań i systemów obliczeniowych.
Pierwszym zadaniem w ramach prac w WP.2.1.6 była więc identyfikacja tych parametrów oraz określenie możliwości technicznych pozyskiwania informacji o wartości tych parametrów zarówno dla zadania jak i systemu obliczeniowego. PCSS przeprowadziło w tym zakresie analizę oraz szereg testów. Miały one na celu określenie jakie parametry systemów obliczeniowych oraz zadań można pobierać i jaki jest ich wpływ na wydajność obliczeń. W analizie i testach rozważano także wpływ działania mechanizmów pobierających dane o wydajności na systemy obliczeniowe i przetwarzanie w systemie NetSolve (dodatkowe obciążenie systemów, dodatkowa komunikacja).

Parametry stanu systemów obliczeniowych

Pierwsza faza analizy dotyczyła możliwości pobierania parametrów stanu systemu obliczeniowego przy użyciu standardowych funkcji systemowych oraz standardowych programów-narzędzi systemu Unix. Skupienie się na standardowych funkcjach i narzędziach uzasadnione jest faktem, że są one dostępne na większości platform Unix'owych, w tym na platformach wykorzystywanych w ramach SGIgrid. Co więcej, funkcje systemowe korzystają z mechanizmów wbudowanych w system operacyjny a przez to zapewniają względnie dużą dokładność pomiarów i wnoszą minimalne narzuty czasowe i zasobowe podczas działania.
Określono jakie parametry można pobierać przy użyciu poszczególnych funkcji systemowych i narzędzi.

Z kolei przeprowadzono analizę dodatkowego obciążenia systemów obliczeniowych spowodowanego przez działanie narzędzi pobierających informacje o wydajności. Przyjęto, że narzut 5% na zużycie zasobów takich jak czas procesora czy pamięć operacyjna jest akceptowalny. Przyjęto, również, że najmniejszą jednostką czasu, dla której pobierane będą parametry wydajnościowe z systemu jest 5 sekund.

Przy takiej częstotliwości próbkowania parametrów narzędzia i funkcje systemowe opisane w Tabeli... wnosiły narzuty na akceptowalnym poziomie, mniejszym niż 5% (stosowane oddzielnie czy symultonicznie?). Testy przeprowadzono na węzłach instalacji testowej (opisanej w pkt. 4 raportu 1 zadania p.n. „Przegląd cech i funkcjonalności systemów udostępniania bibliotek matematycznych, raport z testów i analizy kodów źródłowych, raport z wykonania instalacji testowej").

Parametry zbierane cyklicznie a parametry dotyczące stanu systemu podczas wywołania funkcji:

Mechanizm szeregowania rozwijany w ramach zadania WP.2.1.6 wymaga dostępu do różnego typu danych.
Po pierwsze konieczne jest zbieranie danych potrzebnych do predykcji stanu zasobu obliczeniowego. Predykcja pozwala z pewnym prawdopodobieństwem określić stan zasobów obliczeniowych w pewnym przedziale czasu. Dane te pobierane są cyklicznie i obejmują one obciążenie procesorów, liczbę operacji wejścia/wyjścia, użycie obszaru wymiany, liczbę operacji wymiany, użycie pamięci etc.
Predykcja czasu wykonania danej funkcji obliczeniowej na danym zasobie obliczeniowym wymaga z jednej strony predykcji stanu systemu komputerowego, z drugiej strony znajomości jego charakterystyki wydajnościowej. Charakterystyka określa jaką część zasobów obliczeniowych udostępnia system w danym stanie (szczegóły w punkcie 2.4.1 raportu). Konieczne jest więc zbieranie danych o zasobach zużytych przez wywołanie danej funkcji obliczeniowej. Obejmują one obciążenie procesorów, liczbę operacji wejścia/wyjścia, użycie obszaru wymiany, liczbę operacji wymiany, użycie pamięci etc. Analiza tych danych w powiązaniu z informacjami o stanie zasobów obliczeniowych podczas wyliczania funkcji matematycznej pozwala na rekonstrukcję modelu wydajnościowego maszyny (szczegóły w punkcie 2.3 raportu).

Informacje o zasobach zużytych przez wykonanie funkcji matematycznej i parametry wywołania funkcji matematycznej

Dane o zasobach zużytych przez wykonanie funkcji matematycznej potrzebne są również do wyliczania funkcji pracy danej implementacji funkcji matematycznej (szczegóły w punkcie 2.4.1 raportu). Funkcja pracy określa jaka liczba zasobów potrzebna jest do wyliczenia funkcji obliczeniowej przy danych parametrach jej wywołania (rozmiar problemu, rozmiar danych wejściowych, przewidywany lub znany rozmiar danych wejściowych, parametry modyfikujące działanie algorytmu itp.). Dane dotyczące zasobów zużytych przez wywołanie funkcji obejmują: obciążenie procesorów, liczbę operacji wejścia/wyjścia, użycie obszaru wymiany, liczbę operacji wymiany, użycie pamięci etc. Są one zbierane i analizowane w powiązaniu z danymi dotyczącymi wartości parametrów wywołania funkcji matematycznej. Szczegóły mechanizmu wyliczania funkcji pracy opisane są w punkcie 2.4.1 raportu.

Parametry dotyczące jakości połączenia sieciowego

Ważnym parametrem stanu środowiska obliczeniowego mającego wpływ na czas wyliczania zadania matematycznego jest stan połączenia sieciowego między klientem a węzłem obliczeniowym. Jak wspomniano, NetSolve nie analizuje przepustowości połączenia klient-serwer natomiast monitoruje stan połączenia agent-serwer. Podyktowane jest to faktem, że agent i serwery działają ciągle, podczas gdy proces-klient na komputerze użytkownika uruchamiany jest tylko w momencie wykonywania obliczeń w NetSolve.
Podsystem monitorowania i predykcji wydajności opracowany przez PCSS dla umożliwia predykcję czasu przesyłania danych pomiędzy klientem a serwerem na podstawie informacji o przesyłaniu tych danych w przeszłości.
Predykcja, podobnie jak w przypadku obliczeń opiera się z jednej strony na pobieraniu informacji o przepustowości i opóźnieniu sieci obserwowanej podczas przesyłu danych dla obliczeń, z drugiej strony o pomiar liczby zasobów systemów komputerowych, których te przesyły wymagały. Takie podejście wynika ze stwierdzenia, że przesyły dużych ilości danych przez sieć wymagają z jednej strony dostępności pasma sieciowego, z drugiej strony obciążają zasoby systemu komputerowego (zwiększenie obciążenia CPU, wzrost liczby operacji i/o na sekundę itd.). Dlatego informacje o zużytych przez transmisję zasobach zarówno po stronie klienta jak i po stronie serwera są zbierane w celu późniejszej analizy w module predykcji (szczegóły w punkcie 2.5 raportu).

Parametry dotyczące repozytorium

W zmodyfikowanym NetSolve wprowadzono nowy obiekt – repozytorium – w którym umieszczane są dane wejściowe dla obliczeń i wyniki. Przetwarzanie zlecenia w zmodyfikowanym NetSolve wymaga dwóch operacji

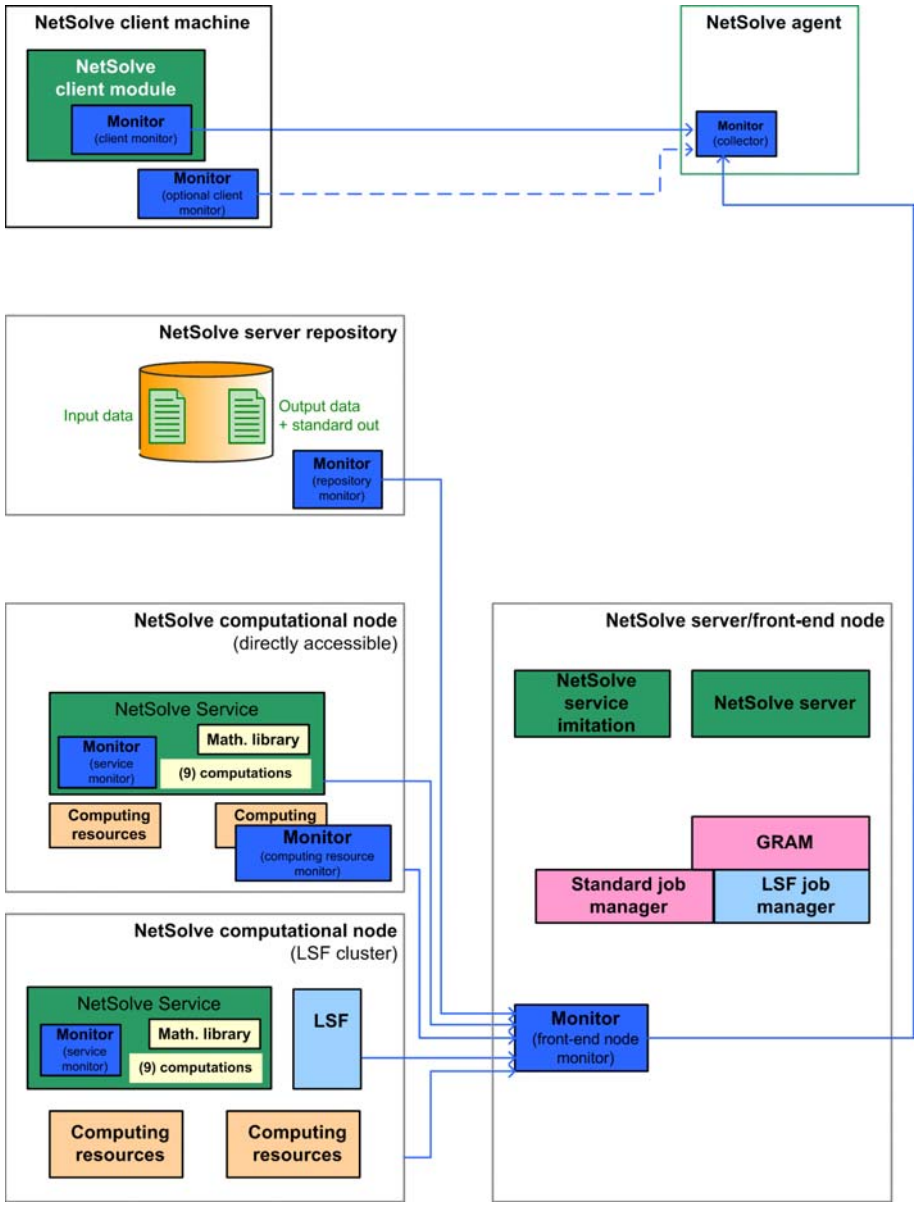
zapisu i dwóch operacji odczytu wykonanych w repozytorium. Moduł klienta umieszcza w repozytorium dane wejściowe (zapis), które są następnie pobierane przez proces-serwis (odczyt). Po zakończeniu obliczeń proces-serwis umieszcza dane wynikowe w repozytorium (zapis) a moduł klienta pobiera je (odczyt). Wydajność zapisu i odczytu w repozytorium ma więc istotny wpływ na wydajność przetwarzania w zmodyfikowanym NetSolve. Dla oceny opłacalności wykonania zadania na danym węźle obliczeniowym konieczne jest uwzględnienie oceny czasu potrzebnego na wykonanie operacji na repozytorium skojarzonym z tym węzłem. Na potrzeby modułu analizy informacji o wydajności podsystem monitorowania pobiera dane dotyczące wydajności operacji na repozytorium oraz dane dotyczące jego stanu. Dane dotyczące operacji na repozytorium obejmują informacje o otrzymywanej prędkości zapisu i odczytu danych oraz o zasobach zużytych na te operacje (czas procesora, liczba operacji wejścia wyjścia, zużyta pamięć operacyjna). Również dane dotyczące stanu zasobów podczas wykonywania operacji na repozytorium są zbierane przez monitor umieszczony w repozytorium. Możliwe jest zbieranie tych danych tylko podczas przetwarzania zleceń w NetSolve lub, opcjonalnie, w sposób ciągły – szczegóły w punkcie 2.2. raportu.

2.2. Opracowanie architektury systemu monitorującego

W punkcie 2.1 omówione są parametry związane z wydajnością przetwarzania w NetSolve, które powinny być zbierane na potrzeby mechanizmu predykcji czasu wykonania i szeregowania zadań. Wyróżnione zostały parametry zbierane okresowo oraz parametry zbierane tylko podczas przetwarzania wywołań funkcji matematycznych.

Architektura systemu monitorowania

Ogólną architekturę systemu zbierania danych wydajnościowych, zwanego dalej systemem monitorowania przedstawia rysunek 1. Elementy systemu rozmieszczone są na węzłach klienta i agenta oraz na węzłach obliczeniowych, w przypadku węzłów dostępnych pośrednio lub na węzłach dostępowych w przypadku zasobów niedostępnych bezpośrednio. Funkcje związane z monitorowaniem włączono również do modułu klienta NetSolve oraz do serwisu.



Rysunek 1 Architektura systemu monitorującego dla NetSolv

Na węźle dostępowym/ serwerowym działa proces-monitor węzła dostępowego (ang. *front-end node monitor*). Zbiera on dane dotyczące stanu węzłów obliczeniowych oraz wydajności obliczeń a także dane dotyczące

Repozytorium skojarzonego z serwerem. W przypadku węzłów obliczeniowych dostępnych bezpośrednio (moduł serwerowy NetSolve działa na tej samej maszynie, na której działa serwis) moduł *front-end node monitor* cyklicznie pobiera dane o stanie systemu obliczeniowego za pomocą odpowiednich funkcji systemowych i programów-narzędzi (omówienie w pkt. 2.1. raportu). W przypadku węzłów obliczeniowych dostępnych pośrednio, proces-monitor cyklicznie odpytuje zarządcę tych zasobów o ich stan. Te dane trafiają do procesu-kolektora w module Agenta (ang. *collector*). Dodatkowo, proces-serwis wykonujący obliczenia (na pośrednio lub bezpośrednio dostępnym węźle obliczeniowym) pobiera informacje dotyczące wykonania funkcji oraz stanu systemu podczas jej wykonania. Dane dotyczące wykonania funkcji obejmują informacje o zasobach zużytych poprzez to wykonanie. Dane dotyczące stanu systemu obliczeniowego podczas obliczeń zbierane przez serwis dotyczą tych samych parametrów stanu systemu, które są monitorowane przez proces-monitor na węźle dostępowym. Takie rozwiązanie przyjęto z uwagi na fakt, że wiązanie oddzielnie zbieranych danych dotyczących wydajności wykonania funkcji z danymi dotyczącymi stanu systemów obliczeniowych nastręczało pewne problemy, spowodowane m.in. przesunięciem czasu pomiędzy węzłami obliczeniowymi oraz trudnością synchronizacji monitorów na węźle dostępowym i obliczeniowym (w przypadku, gdy są to oddzielne węzły). Rekonstrukcja modeli wydajności zasobów obliczeniowych jest tym lepsza im lepiej skorelowane są te pomiary.

Monitor działajacy w repozytorium (ang. *repository monitor*) monitoruje wydajność operacji zapisu i odczytu do i z repozytorium oraz kontroluje obciążenie zasobów (dyski, CPU itd.). Funkcję monitorowania wydajności operacji zapisu i odczytu danych zaimplementowano poprzez dołączenie do kodu serwera GridFTP funkcji pobierających i logujących informacje o wydajności. Informacje te pobierane są podczas przetwarzania zleceń NetSolve. Dołączony do serwera GridFTP kod pobiera wartości parametrów stanu systemu komputerowego, na którym znajduje się repozytorium obejmujące stan procesorów, wielkość dostępnej i zajętej pamięci operacyjnej oraz liczba operacji wejścia/wyjścia na sekundę. Pomiary stanu tych zasobów ograniczone są zatem do okresów czasu, w których przetwarzane są zlecenia NetSolve. Możliwe jest także wykorzystanie oddzielnego procesu-monitora, działającego ciągle, niezależnie od faktu, czy w danym momencie przetwarzane są zlecenia NetSolve. W tym przypadku możliwa jest predykcja czasu potrzebnego na wykonanie operacji na repozytorium na podstawie statystyk dotyczących poprzednich operacji oraz bieżącego stanu systemu. Bez procesu-monitora dla repozytorium predykcja czasu zapisu/odczytu danych do/z repozytorium odbywa się wyłącznie na podstawie historycznych danych. Używanie oddzielnego procesu monitora sensowne jest jedynie w przypadku, gdy repozytorium znajduje się na innym systemie komputerowym niż moduł serwerowy NetSolve, ponieważ monitor działający na potrzeby węzła serwerowego kontroluje stan wszystkich elementów systemu komputerowego, które są istotne dla działania repozytorium (stan procesorów, pamięci i operacji wejścia/wyjścia).

Na maszynie klienta mogą działać dwa rodzaje monitorów. W kodzie klienta NetSolve zaszyto funkcje logujące informacje o wartości parametrów zadania a także informacje o wydajności przesyłu danych do i z repozytorium. Te informacje są następnie przesyłane do procesu-kolektora w agencie NetSolve. Używane są przy szeregowaniu kolejnych wywołań funkcji matematycznych przez mechanizm analizy danych historycznych. Opcjonalnie na maszynie klienta może działać również proces-monitor, który cyklicznie testuje przepustowość i opóźnienie sieci pomiędzy modułem klienta a serwerami obliczeniowymi (ang. *optional client monitor*). Uzyskane w ten sposób informacje o stanie sieci przesyłane są do procesu-kolektora w Agencie NetSolve i używane do predykcji czasu przesyłana danych wejściowych i wyjściowych dla obliczeń. Listę serwerów do monitorowania proces-monitor uzyskuje z Agenta NetSolve poprzez wydanie odpowiedniego zapytania zgodnego z protokołem NetSolve-over-GlobusIO.
Możliwe jest też uruchamianie modułu serwerowego NetSolve na węzłach klienckich. Taka opcja używana jest dla mechanizmu warunkowego wykonania lokalnego opracowywanego w ramach zadań WP.2.1.3 i WP.2.1.4. W tym wypadku monitorowanie stanu węzła klienckiego odbywa się takimi samymi metodami jak monitorowanie stanu węzła serwerowego.

Implementacja systemu monitorowania

Pobieranie danych o stanie systemów oraz wydajności obliczeń w NetSolve wykonywane jest przy użyciu funkcji systemowych oraz programów-narzędzi (opis w pkt. 2.1 raportu). Przesyłanie danych do procesu-kolektora w Agencie NetSolve odbywa się przy użyciu systemu *NetLogger* [NetLogger1][NetLogger2]. System *NetLogger* pozwala na przesyłanie komunikatów oznaczonych stemplem czasowym oraz identyfikatorem procesu, który go wysyła do procesu-kolektora. Komunikaty mają format „zmienna=wartość”. Informacje o wydajności przetwarzania w NetSolve oraz o stanie systemów obliczeniowych przesyłane są do procesu-kolektora w tym formacie.
Wykorzystanie funkcjonalności systemu NetLogger wymaga dołączenia biblioteki systemu NetLogger do modułów serwerowych, serwisu, serwera GridFTP w repozytorium, opcjonalnego modułu monitorującego w repozytorium oraz do modułu klienta i modułu opcjonalnego monitora klienta. Podobnie, Agent musi być

wyposażony w tę bibliotekę. W zmiennych środowiskowych poszczególnych maszyn ustawiony musi być adres procesu-kolektora danych.

Funkcjonalność systemu monitorowania

Opracowany w PCSS system monitorowania dla NetSolve umożliwia zbieranie informacji o przetwarzaniu w NetSolve. Podstawową funkcją jest monitorowanie stanu węzłów i wydajności przetwarzania w NetSolve na potrzeby szeregowania zadań. Dane zbierane cyklicznie i dane związane z wywołaniem funkcji przetwarzane są przez moduł analizy informacji wydajnościowych (więcej szczegółów w pkt. 2.3 raportu). Na ich podstawie wyliczane są funkcje pracy implementacji funkcji matematycznej i modele wydajnościowe serwerów obliczeniowych oraz wykonywana predykcja stanu tych zasobów.
Możliwa jest również wizualizacja danych o wydajności przetwarzania w NetSolve. Dotyczy to zarówno nieprzetworzonych danych o wydajności (wykresy obciążenia zasobów w czasie), jak i wyników analizy (wykresy zrekonstruowanych funkcji). Możliwa jest również wizualizacja przebiegu wykonania poszczególnych zdalnych wywołań funkcji matematycznych w NetSolve. Ułatwia ona użytkownikom oraz administratorowi systemu analizę przebiegu przetwarzania zadań, m.in. obrazując w sposób graficzny etapy przetwarzania funkcji i ilość czasu spędzoną przez zlecenia w poszczególnych stadiach przetwarzania. Taka analiza możliwa jest dla pojedynczych wywołań funkcji lub dla wszystkich wywołań funkcji danego użytkownika lub instancji NetSolve.
Dla realizacji wspomnianej funkcjonalności konieczne było opracowanie mechanizmu post-processingu informacji o przetwarzaniu zadania. Postprocessing ma na celu powiązania ze sobą operacji wykonanych w ramach obsługi konkretnego zadania zdalnego uruchomienia funkcji wykonanych w różnych modułach NetSolve.

Post-processing informacji o wydajności przetwarzania

Moduł klienta NetSolve oraz Proxy NetSolve posługują się we wstępnej fazie przetwarzania zlecenia identyfikatorem *request_id* do identyfikacji przetwarzanego przez nie zlecenia. Ten identyfikator używany jest aż do momentu przydzielenia przez Agenta *global_id* dla zadania podczas obsługi żądania o przydział zasobów. Śledzenie przebiegu wszystkich stadiów przetwarzania w NetSolve wymaga powiązania *request_id* z *global_id*. W tym celu dane zbierane przez proces-kolektor w Agencie są okresowo przeglądane. Najpierw wynajduje się informacje o przydzieleniu *global_id* do danego *request_id* a następnie zastępuje wpisy zawierające tylko *request_id* oraz wpisy zawierające tylko *global_id* wpisami zawierającymi zmieniony, unikalny identyfikator zlecenia („suma” *request_id* i *global_id*). Śledzenie przebiegu zlecenia NetSolve w systemie możliwe jest przy użyciu tego zmienionego identyfikatora.

2.3. Badania wydajnościowych modeli przetwarzania maszyn o różnych architekturach i modeli przetwarzania systemów obliczeniowych pracujących pod kontrolą zarządców zasobów

Prace na badaniem modeli wydajnościowych przetwarzania maszyn oraz modeli przetwarzania systemów obliczeniowych objęły m.in. analizę modelu wydajności zasobów obliczeniowych stosowanego w oryginalnym NetSolve oraz model stosowany w systemie RCS (*ang. Remote Computation Service* [RCS1] [RCS2]. Modele te rozważano ze względu na fakt, że zostały zaprojektowane dla środowisk, w których mechanizm przetwarzania zleceń jest podobny.
Modele te zostaną pokrótce omówione oraz skomentowana zostanie ich użyteczność dla szeregowania danych w zmodyfikowanym NetSolve.

2.3.1. Model wydajności zasobów w oryginalnym NetSolve

Autorzy systemu *NetSolve* skonstruowali prosty, teoretyczny model pozwalający na estymację wydajności przy danej „surowej" wydajności i obciążeniu. Ten model daje estymowaną wydajność p, jako funkcję obciążenia w, „surowej" wydajności P, oraz liczby procesorów na maszynie n:

$$p = \frac{P \times 100 \times n}{100 \times n + \max(w - 100 \times (n-1), 0)}.$$

Zdaniem autorów NetSolve model ten sprawdza się w praktyce i jest bliski wynikom eksperymentów. Model nie uwzględnia jednak narzutów systemu operacyjnego na migrację procesów w przypadku, gdy maszyna, na której wykonywane są obliczenia jest wieloprocesorowa. Natomiast dobrze obrazuje on np. fakt, że spadek wydajności obliczeń w takiej konfiguracji sprzętowej w stosunku do „surowej" wydajności maszyny zaczyna się dopiero po obciążeniu wszystkich procesorów.

2.3.2. Model wydajności stosowany w systemie RCS

System RCS [RCS1] [RCS2] (ang. Remote Computation System – system zdalnego liczenia) udostępnienia dużą mocy obliczeniowej użytkownikom w taki sposób, by nie musieli oni zajmować się nisko-poziomowymi szczegółami związanymi z architekturą poszczególnych serwerów obliczeniowych. Użytkownik widzi system RCS jako zwykłą bibliotekę programową, której funkcje mogą być wywoływane z lokalnej maszyny. Obliczenia wykonywane są na dynamicznie przydzielonej, należącej do pewnej puli komputerów maszynie. Przydział odbywa się w taki sposób, by zminimalizować czas obliczeń. W tym sensie sposób przetwarzania w systemie RCS jest zbliżony do NetSolve. Jednakże poważnym ograniczeniem RCS jest fakt, że oparty jest on o system PVM (*ang. Parallel Virtual Machine*), więc działa tylko w obrębie danej wirtualnej maszyny obliczeniowej. Poza tym rozwiązuje tylko problemy algebry liniowej (używa, po stronie maszyn obliczeniowych pakietu LAPACK) i nie jest przystosowany do rozwiązywania innych problemów matematycznych.
Autorzy systemu RCS używają modelu czasu odpowiedzi T jako funkcji parametrów takich jak rozmiar problemu n, liczba procesorów p i inne charakterystyki algorytmu i sprzętu:

$$T = f(n, p, ...).$$

Ich zdaniem, przez wzgląd na sposób działania systemu RCS, czas odpowiedzi powinien być obliczany na podstawie trzech składowych: czasu uruchamiania modułu obliczeniowego T_s (ang. *solver start up time*), czasu transmisji T_t i czasu obliczeń T_c.

Problemy modelowania czasu obliczeń zauważone przez autorów systemu RCS

Dla aplikacji obliczeniowych algebry liniowej wskaźnik liczby operacji obliczeniowych (flop) zdaniem autorów systemu RCS dobrze określa ilość „pracy", którą trzeba wykonać dla rozwiązania problemu. Czas wykonania (ang. *execution time*), z kolei, jest współczynnikiem pomiędzy tą liczbą pracy a prędkością obliczeniową komputera.
Zdaniem autorów systemu RCS, przedstawione podejście napotyka na problemy. Po pierwsze liczba pracy do wykonania jest często zależna od samego problemu obliczeniowego. Liczba operacji może być określona jako funkcja rozmiaru problemu jedynie w przypadku metod dokładnych. Dla oceny wielkości pracy metod iteracyjnych można stosować jedynie metody oparte o przybliżenie współczynnika zbieżności tych algorytmów.
Autorzy RCS uważają, że metoda estymacji czasu obliczeń opartą na ekstrapolacji otrzymanych na drodze eksperymentalnej wyników pomiarów również nie jest najlepsza. Wymaga ona, po pierwsze istnienia modelu

czasu wykonania jako funkcji rozmiaru problemu. Nieznane współczynniki takiej funkcji złożoności obliczeniowej dobierane by były poprzez interpolację, np. metodą najlepszego kwadratowego dopasowania. Jednakże niebezpieczeństwem takiego podejścia, zdaniem autorów systemu RCS jest fakt, iż pomiary pozwalają określić wydajność jedynie w wąskim zakresie, być może wielowymiarowej, przestrzeni parametrów wpływających na wydajności (ma to szczególnie dotyczyć algorytmów uruchamianych na wysoko-wydajnych komputerach).

Model proponowany przez autorów RCS

Autorzy systemu RCS proponują modele dla oceny czasu obliczeń, z których każdy odzwierciedla cechy pewnej klasy systemów obliczeniowych.

Dla komputerów konwencjonalnych proponują model czasu obliczeń, który można wyrazić za pomocą następującej formuły:

$$T = \frac{c(n)}{r},$$

gdzie $c(n)$ wyraża liczbę operacji zmienno-przecinkowych zależną od rozmiaru problemu *n* natomiast *r* oznacza prędkość obliczeniową. Parametr *r* jest zależny od maszyny i od algorytmu i jest aproksymowany przez wstawianie wyników kilku pomiarów do równania.

Czas wykonania jednej operacji zmiennoprzecinkowej jest odwrotnością prędkości obliczeń i jest oznaczany jako τ:

$$\tau = 1/r.$$

Dla komputerów o dużej wydajności ze współdzieloną pamięcią autorzy systemu RCS proponują inny model. Zauważają, że techniki stosowane we współczesnych komputerach (przetwarzani potokowe, wektorowe, super-skalarne, hierarchiczność pamięci, wieloprocesorowość itd.) sprawiają, że prędkość obliczeniowa (tzn. praca dzielona przez czas), nie jest stałą wartością. Obserwowany współczynnik wykonania *r*, w ich opinii zależy od algorytmu i rozmiaru problemu i jest dobrze przybliżany przez model Hockney'a i Jesshope'a:

$$r(n) = \frac{r_\infty}{\frac{n_{1/2}}{n} + 1}.$$

Wydajność *r* jest wyrażona w zależności od parametrów r_∞ i $n_{1/2}$. Te parametry są charakterystyczne dla określonego algorytmu i oznaczają szczytową wydajność dla problemów o dużym rozmiarze (r_∞) i rozmiar problemu ($n_{1/2}$), przy którym osiągana jest połowa szczytowej wydajności.

Tak więc, prosty model dla czasu obliczeń algorytmu działającego na sekwencyjnym komputerze lub komputerze wieloprocesorowym ze współdzieloną pamięcią, zdaniem autorów systemu RCS, wygląda następująco:

$$T_c = \frac{c(n)}{r(n,p)},$$

gdzie $c(n)$ jest liczbą operacji zmiennoprzecinkowych dla algorytmu a $r(n,p)$ jest współczynnikiem wykonania na określonym komputerze, zależnym od rozmiaru problemu i liczby procesorów. Dla klasycznych sekwencyjnych komputerów $r(n)$ na ogół nie zależy od *n*.

2.3.3. Przydatność istniejących modeli wydajności

W praktyce użycie żadnego z opisanych powyżej modeli nie gwarantuje odpowiednio duże dokładności przewidywania czasu wykonania zadań i sprawnego rozdziału zasobów w zmodyfikowanym NetSolve. Wynika to z szeregu faktów. Model opracowany dla oryginalnego NetSolve sprawdza się na węzłach obliczeniowych złożonych z pojedynczych maszyn, na których nie zachodzą zjawiska zrównoleglenia wykonania kodu binarnego. Model ten nie uwzględnia też wielu czynników mających wpływ na wydajność obliczeń w systemie komputerowym takich jak stan elementów systemu innych niż procesory. Nie jest również dostosowany do środowisk takich jak np. systemy kolejkowe, które mogą być przez zmodyfikowany NetSolve używane jako zasoby obliczeniowe.

Model opracowany przez autorów systemu RCS ma kilka ciekawych własności. Przede wszystkim uwzględnia zjawisko zmiennej prędkości obliczeniowej, rozumianej jako praca wyrażona w liczbie operacji zmienno-przecinkowych dzielona przez czas, zachodzące w nowoczesnych systemach komputerowych.

2.3.4. Teoretyczne a rekonstruowane modele wydajności

Poza omówionymi modelami wydajności PCSS analizowało również prace innych autorów dotyczące przewidywania czasu wykonania zadań obliczeniowych w systemie komputerowym. W wyniku analizy tych prac PCSS stwierdziło, że na potrzeby zmodyfikowanego systemu NetSolve najlepsze będzie używanie modeli wydajności, które są dynamicznie tworzone na podstawie danych o przebiegu przetwarzania w danym środowisku. Stwierdzenie to wynika z faktu, że nie ma statycznych modeli wydajności, które sprawdzają się dobrze w każdym środowisku i na każdym typie zasobów obliczeniowych (komputery skalarne, wektorowe, klastry itd.).

PCSS opracowało mechanizm analizy informacji o wydajności i rekonstrukcji na tej podstawie modelów wydajności zasobów biorących udział w przetwarzaniu w NetSolve a także wyciągania wniosków co do wydajnościowej charakterystyki urchamianych w NetSolve zadań. W ramach rekonstrukcji funkcji modelujących charakterystyki wydajnościowe zasobów i zadań podejmowane są próby dopasowania współczynników funkcji modelujących opartych o dobrze znane modele wydajności zasobów i zadań (m.in. opisane w tym punkcie). Szczegółowy opis metod analizy informacji o wydajności przetwarzania zawiera punkt 2.4. raportu.

2.4. Opracowanie metod automatycznej analizy danych

Ocena czasu wykonania zdalnego wykonania funkcji obliczeniowej opiera się przede wszystkim na ocenie czasu obliczeń składających się na rozwiązanie zadania matematycznego na zdalnym zasobie. Uwzględnia też czynniki takie jak czas przesyłu danych wejściowych i wyjściowych oraz czas zapisu i odczytu tych danych do i z repozytorium.
W pierwszej kolejności omówiona zostanie automatyczna analiza danych dotyczących wydajności mająca na celu odtworzenie z danych historycznych funkcji pracy implementacji problemów matematycznych oraz funkcji określających modele wydajności zasobów obliczeniowych (punkt 2.3.1).
Analiza danych historycznych dotycząca informacji o wydajności operacji na repozytorium oraz informacji o wydajności przesyłów danych wejściowych i wyjściowych dla obliczeń pomiędzy modułem klienta i serwera NetSolve odbywa się na analogicznych zasadach jak analiza danych dotyczących obliczeń. Omówiona została w punkcie 2.3.2 raportu.

2.4.1. Rekonstrukcja funkcji pracy i funkcji modelu wydajności dla obliczeń

PCSS opracowało metodę rekonstrukcji funkcji pracy implementacji funkcji matematycznej oraz funkcji modeli wydajności zasobów. Dla omówienia metod rekonstrukcji zostaną zdefiniowane pewne pojęcia.

Definicja pojęć

Instancja $T_{n,OP}$ problemu P to zadanie wykonania funkcji matematycznej. n_T jest *rozmiarem* tej instancji. OP_T zawiera *inne parametry* wywołania funkcji matematycznej (np. wersja algorytmu, parametry modyfikujące działanie algorytmu). Zdefiniujmy również *pracę W_{CTR}*, która musi być wykonana dla rozwiązania instancji *T* problemu obliczeniowego na *systemie obliczeniowym C* przy użyciu *zasobu R*. Praca określona jest jako ilość *zasobu R* potrzebna do rozwiązania zadania. Zasobem może być np. ilość czasu, którą musi spędzić proces obliczeniowy w trybie użytkownika i trybie jądra na maszynie *C* (czas procesora, nie czas rzeczywisty). Praca może być też wyrażona jako ilość innego z zasobów systemu komputerowego, np. wielkość użytej pamięci operacyjnej, liczba operacji wejścia/wyjścia potrzebnych do rozwiązania zadania itp.
Zdefiniujmy również *funkcję pracy*, która obrazuje zależność pomiędzy rozmiarem n_T i innymi parametrami OP_T instancji $T_{n,OP}$ problemu *T* i pracą W_{CTR}, która musi być wykonana w systemie obliczeniowym *C* przy użyciu zasobu *R*.

$$W_{CTR} = f_{CTR}(n_T, OP_T)$$

PCSS zdecydowało się używać funkcji pracy zamiast funkcji złożoności obliczeniowej jako miary pracy, która musi być wykonana dla rozwiązania zadania obliczeniowego z kilku powodów. Po pierwsze, cechy funkcji złożoności obliczeniowej wykluczają automatyzację procesu jej rekonstrukcji. Nie jest możliwe zmierzenie liczby operacji elementarnych wykonanych w ramach rozwiązywania danego zadania matematycznego na zasobie obliczeniowym. Zużycie zasobów takich jak czas procesora, pamięć, operacje wejścia/wyjścia itd. jest mierzalne za pomocą narzędzi systemu operacyjnego. Dzięki temu możliwa jest rekonstrukcja funkcji pracy z danych historycznych dotyczących wykonania zadania na danym zasobie.
Po drugie, funkcja złożoności obliczeniowej jest miarą wysoko-poziomową. Określa ona liczbę kroków algorytmu rozwiązującego dany problem na podstawie rozmiaru zadania. Kroki algorytmu są jednoznaczne z operacjami na wysokim poziomie abstrakcji. Tymczasem optymalizacja kodu binarnego wykonywana przez kompilatory a także rozmaite techniki zrównoleglenia wykonywania kodu binarnego stosowane w nowoczesnych systemach komputerowych mogą sprawiać, że zależność między rozmiarem zadania a liczbą zasobów (rozumianą np. jako czas procesora) potrzebnych do jego rozwiązania jest odmienna od teoretycznej funkcji złożoności obliczeniowej.
Zdefiniujmy również funkcję g_C modelu wydajnościowego maszyny *C*. Określa ona zależność pomiędzy stanem S_C maszyny *C*, ilością danego zasobu W_R (pracą wykonaną na danym zasobie) a ilością czasu zegarowego $time_{WR}$ potrzebną dla uzyskania tej ilości zasobu przez proces obliczeniowy na maszynie *C* w stanie S_C.

$$time_{W_R} = g_{CR}(W_R, S_C)$$

Funkcja g_C obrazuje politykę zarządzania zasobami w danym systemie obliczeniowym. Wartości i argumenty tej funkcji są mierzalne za pomocą narzędzi systemu operacyjnego. Daje to możliwość jej rekonstrukcji na podstawie danych historycznych dotyczących wydajności obliczeń w danym systemie komputerowym.
Stan S_C systemu obliczeniowego to zbiór parametrów określających stan poszczególnych zasobów systemu. Obejmują one parametry statyczne: liczba procesorów *I*, ilość pamięci fizycznej *PM* itd. oraz parametry

dynamiczne takie jak: obciążenie L systemu, ilość dostępnej pamięci M, intensywność operacji wejścia/wyjścia I oraz inne parametry związane z wydajnością. Wartości tych parametrów mogą być pozyskane w systemach operacyjnych z rodziny *Unix* za pomocą funkcji i narzędzi systemowych.
Ważną cechą funkcji modelu wydajności jest fakt, iż uwzględnia ona wpływ stanu wielu zasobów systemu obliczeniowego na dostępność danego zasobu. Ma to znaczenie praktyczne. W rzeczywistych systemach komputerowych zdarzają się sytuacje, w których proces obliczeniowy nie może skorzystać z jednego z zasobów (np. czas procesora), mimo że jest on dostępny, z powodu przeciążenia innego z zasobów (np. podsystemu wejścia/wyjścia) biorących udział w przetwarzaniu.

Gdy rozpatrujemy konkretną instancję $T_{n,OP}$ problemu matematycznego P, której rozwiązanie wymaga pewnej ilości pracy W_{CTR} na danym zasobie (pewnej ilości zasobu R) otrzymujemy zależność:

$$time_{W_{CTR}} = g_{CR}(W_{CTR}, S_C).$$

Z tej zależności wynika, że ocena czasu zegarowego potrzebnego do rozwiązania danej instancji $T_{n,OP}$ problemu obliczeniowego P wymaga znajomości ilość pracy do wykonania W_{CTR} oraz znajomości stanu zasobu S_C podczas rozwiązywania zadania. Wyliczenie wielkości pracy W_{CTR} wymaga znajomości wartości parametrów n_T i OP_T danej instancji $T_{n,OP}$ problemu i użycia funkcji pracy f_{CTR}. Wartości elementów zbioru S_C są *a priori* nieznane.

Mogą być przewidywane z pewnym prawdopodobieństwem przy użyciu np. zewnętrznych systemów prognostycznych takich jak Network Weather Service [NWS1] [NWS2]. Oczywiste jest, że dokładność oceny czasu potrzebnego na rozwiązanie zadania ograniczona jest dokładnością metod predykcji. W opracowanym przez PCSS mechanizmie wykorzystuje się funkcje systemu NWS do predykcji stanu procesorów i pamięci oraz przepustowości i opóźnienia obserwowanego na połączeniach sieciowych. W przypadku pozostałych zasobów używa się metody „LAST", tzn. zakłada się, że stan tych zasobów będzie taki, jak wskazuje ostatnio dostępny pomiar stanu tego zasobu pochodzący z systemu monitorowania.

Ocena czasu zegarowego potrzebnego do rozwiązania danej instancji $T_{n,OP}$ problemu obliczeniowego P na danym zasobie obliczeniowym wymaga znajomości formy i współczynników funkcji pracy f_{CTR} i funkcji modelu wydajnościowego g_{CR} poszczególnych zasobów systemu komputerowego.

Przyjmujemy założenie, że funkcja pracy i funkcja modelu wydajności analizowane będą dla ograniczonego zakresu zasobów systemu operacyjnego. W implementowanym w ramach zadania WP.2.1.6 projektu celowego rozwiązaniu pod uwagę będą brane: czas procesora, wielkość dostępnej i używanej pamięci operacyjnej, liczba operacji wejścia/wyjścia, wielkość obszaru wymiany (ang. *swap*) i liczba operacji wymiany. Funkcje modelu wydajności poszczególnych zasobów uwzględniają jedynie stan wymienionych zasobów, np. funkcja modelu wydajności zasobu „czas procesora" uwzględnia stan pamięci, operacje wejścia/wyjścia w systemie, wielkość obszaru wymiany i liczę operacji wymiany na sekundę.

Rekonstrukcja funkcji

Forma i współczynniki funkcji pracy f_{CTR} oraz funkcji modelu wydajności g_{CR} zasobu nie są znane *a priori*. Mogą być one zrekonstruowane przy użyciu danych eksperymentalnych dotyczących wywołań danej funkcji matematycznej na danym zasobie. Funkcja pracy f_{CTR} może być odtworzona po zebraniu odpowiedniej liczby pomiarów dotyczących parametrów n_T i OP_T poszczególnych instancji problemu matematycznego i wielkości pracy W_{CTR} potrzebnych do rozwiązania tych instancji problemu w danym systemie komputerowym. Funkcja modelu wydajności g_{CR} zasobu może być odtworzona na podstawie odpowiednio dużej liczby pomiarów wielkości pracy do wykonania W_{CTR} oraz ilości czasu zegarowego potrzebnych do wykonania tych wielkości pracy na zasobie.
Rekonstrukcja funkcji ma dwie fazy. W pierwszej fazie, dla danej funkcji (pracy lub modelu wydajności) „odsiewane" są argumenty statyczne, tzn. takie, których wartości nie zmieniają się. Następnie wykrywane są argumenty, które nie mają wpływu na wartość funkcji. Za takie uznaje się argumenty, dla których współczynnik korelacji z wartością funkcji jest mniejszy niż pewna zdefiniowana wartość (domyślnie mniejszy niż 0,1 i większy niż –0,1). Te argumenty nie są uwzględniane w danym przebiegu rekonstrukcji funkcji. W drugiej fazie rekonstrukcji podejmowana jest próba odtworzenia formy i współczynników funkcji określonej na argumentach, które uznane zostały za znaczące. PCSS rozważało różne metody rekonstrukcji funkcji. W rozwiązaniu opracowanym na potrzeby projektu celowego rekonstrukcja polega na próbie dopasowania współczynników kilku szablonowych funkcji. Szablony te wprowadzone są do konfiguracji modułu analizy wydajności przez administratora systemu (dystrybucja zmodyfikowanego NetSolve zawiera kilka przykładowych szablonów) i interpretowane podczas działania modułu analizy informacji o wydajności. Przykładowe szablony funkcji zawierają funkcje wielomianowe, funkcje liniowe i wykładnicze. Po zakończeniu próby dopasowania współczynników funkcji dla każdej ze zrekonstruowanych funkcji określana jest miara dokładności dopasowania funkcji (średni błąd kwadratowy). Zrekonstruowana funkcja jest uznawana za wiarygodną, gdy jej współczynnik

dopasowania jest większy niż pewna wartość zdefiniowana w konfiguracji modułu analizy informacji o wydajnosci (domyślnie: średni błąd kwadratowy mniejszy niż 10%, średni błąd ... mniejszy niż). Uznana za wiarygodną funkcja pracy lub modelu wydajności jest używana do predykcji czasu wykonania obliczeń na danym zasobie.

Implementacja mechanizmu

Pomiary poszczególnych argumentów i wartości funkcji pracy i funkcji modelu wydajności w danym środowisku obliczeniowym można wykonać na drodze eksperymentalnej. Odpowiednio duża seria testowych obliczeń środowiska pozwala na zebranie odpowiednio dużych kolekcji wartości poszczególnych parametrów. W praktyce koszt takiego „uczenia" jest jednak nieakceptowalny. Dlatego PCSS opracowało koncepcję rekonstrukcji funkcji pracy i funkcji modelu wydajności na podstawie danych historycznych dotyczących wydajności zbieranych podczas działania systemu NetSolve. W początkowej fazie działania NetSolve w danym środowisku obliczeniowym stosowane są metody szeregowania z oryginalnego NetSolve. Równocześnie zbierane są dane dotyczące wydajności przetwarzania i podejmowane próby rekonstrukcji funkcji pracy dla poszczególnych implementacji funkcji matematycznych i funkcji modelu wydajności poszczególnych zasobów. Rekonstrukcji funkcji dokonuje się po zebraniu pewnej minimalnej (domyślnie 10) liczby punktów (argumentów i wartości). Rekonstrukcja funkcji jest wykonywana każdorazowo po uzyskaniu kolejnego kompletu argumentów i wartości dla danej funkcji. Zapewnia to aktualizację formy i współczynników funkcji wraz ze zmieniającymi się cechami środowiska. Nakład pracy związany z każdorazową rekonstrukcją funkcji pracy i funkcji modelu wydajności jest zdaniem PCSS opłacalny. Pozwala on bowiem zabezpieczyć się przed sytuacją, gdy zrekonstruowana nieaktualna funkcja pracy lub modelu wydajności powoduje niepoprawne działanie mechanizmu szeregowania.

Zrekonstruowane funkcje używane są do predykcji czasu wykonania zadania obliczeniowego na zasobach wraz z predykcjami wykonywanymi oryginalnymi metodami NetSolve. Ważność predykcji wykonanej zrekonstruowaną funkcją jest punktowana odpowiednią wartością. Wartość ta jest tym większa, im więcej punktów dla danej funkcji udało się zebrać. Przewidywana wartość czasu wykonania obliczana jest zgodnie z formułą:

$$time = \frac{1 * NetSolve\_time + w * PSNC\_time}{1 + w},$$

gdzie *time* jest przewidywanym czasem wykonania obliczeń, *NetSolve_value* jest wartością przewidywaną metodami oryginalnego NetSolve, natomiast *PSNC_time* jest wartością czasu wykonania przewidywaną metodą opracowaną przez PCSS.
Wartość wagi prognozy dotyczącej czasu wykonania obliczeń wykonanej metodami opracowanymi przez PCSS jest dodatkowo modyfikowana przez mechanizm analizy wydajności w trakcie jego działania w zależności od celności prognoz wykonanych tą metodą. Każdorazowo (lub co zdefiniowany w konfiguracji mechanizmu analizy okres czasu lub co liczbę użyć danej implementacji funkcji matematycznej na danym zasobie) po zakończeniu obliczeń przewidywana i faktyczna wielkość czasu potrzebnego na rozwiązanie zadania są porównywane i wyliczany jest średni błąd kwadratowy oraz średni błąd ... predykcji wykonanych przy użyci zrekonstruowanych funkcji. Porównanie prognoz i wyników dotyczy okresu czasu lub liczby prognoz określonego w konfiguracji modułu analizy danych. Im mniejszy jest wyliczony błąd predykcji, tym większa jest wartość wagi *w* określającej ważność predykcji czasu wykonania wykonywanej metodą opracowaną przez PCSS dla następnych szeregowanych zleceń.
Ważenie prognozy wykonanej metodą opracowaną PCSS dotyczącej czasu rozwiązywania problemu matematycznego ma na celu uniknięcie wykorzystywania metody PCSS, gdy w danym środowisku uzyskuje ona złą dokładność. Kolejną metodą weryfikacji poprawności działania mechanizmu analizy informacji o wydajności opracowanego przez PCSS jest wizualizacja wyników analizy. Mechanizm analizy dostarcza dane dla podsystemu wizualizacji (szczegóły w punkcie 2.5 raportu) dotyczące zrekonstruowanych funkcji pracy oraz modeli wydajnościowych zasobów. Z kolei mechanizm wizualizacji generuje wykresy tych funkcji. Wykresy ułatwiają administratorowi danej instalacji systemu NetSolve weryfikację wyników działania mechanizmu analizy i ewentualną korektę jego ustawień.

2.4.2. Rekonstrukcja funkcji pracy i funkcji modelu wydajności dla operacji na repozytorium oraz przesyłów przez sieć

Ocena czasu wykonania zdalnego wykonania funkcji obliczeniowej oprócz oceny czasu obliczeń potrzebnych do rozwiązania zadania matematycznego na zdalnym zasobie uwzględnia również czynniki takie jak czas przesyłu danych wejściowych i wyjściowych przez sieć oraz czas zapisu i odczytu tych danych do i z repozytorium.
Analiza danych historycznych dotycząca informacji o wydajności operacji na repozytorium oraz informacji o wydajności przesyłów danych wejściowych i wyjściowych dla obliczeń pomiędzy modułem klienta i serwera

NetSolve odbywa się metodami analogicznymi do analizy danych dotyczących obliczeń (omówionych w punkcie 2.3.1 raportu).
Operacje zapisu i odczytu danych w repozytorium oraz przesyły sieciowe pomiędzy modułem klienta a repozytorium są monitorowane przez osadzenie w kodzie serwera GridFTP obsługującego repozytorium instrukcji zbierających dane o wydajności tych operacji. Zbierane dane obejmują informacje o uzyskiwanej prędkości transmisji sieciowejoraz prędkości zapisu i odczytu danych na dyski repozytorium. Obejmują one również konsumowane zasoby: czas procesora, pamięć operacyjna, operacje wejścia/wyjścia oraz użycie obszaru wymiany). Informacja o stanie wybranych zasobów (procesory, pamięć operacyjna, stan obszaru wymiany, ogólna liczba operacji wymiany i operacji wejścia/wyjścia w systemie komputerowym, stan sieci na połączeniu klient-repozytorium) są również pobieranea podczas operacji w repozytorium przez kod osadzony w serwerze. Dane te zbierane są wyłącznie podczas przetwarzania zleceń NetSolve. Używane są przez moduł analizy informacji o wydajności.
Podobnie jak w przypadku obliczeń definiujemy funkcję pracy operacji na repozytorium. Funkcja pracy określa zależność pomiędzy ilością zapisywanych lub odczytywanych oraz przesyłanych przez sieć danych a ilością zasobów potrzebną do wykonania tych czynności. Rozpatrywane zasoby obejmują operacje wejścia/wyjścia, czas procesora, operacje na obszarze wymiany, pamięć operacyjną oraz dostępne pasmo sieciowe i opóźnienie w sieci. Ilość danych do zapisu i odczytu i do przesłania przez sieć rozpatrywanych przez funkcję pracy dla repozytorium i przesyłów sieciowych jest wyrażona w bajtach. Te wielkości $input_T$ i $output_T$ są znane w momencie szeregowania zadania zdalnego wywołania funkcji. Wielkość pracy na danym zasobie R konieczną do wykonania operacji na repozytorium wylicza się przy użyciu dwóch funkcji pracy: jednej dla zapisu i jednej dla odczytu danych. Użycie oddzielnych funkcji pracy dla operacji zapisu i odczytu wiąże się z faktem, że w rzeczywistości zapis danych wymaga z reguły większej ilości zasobu niż odczyt, potrzebnych m.in. na tworzenie odpowiednich struktur w systemie plików. Wyliczanie ilości pracy $W_{TRWrite}$ i W_{TRRead} koniecznych do wykonania zapisów i odczytów danych zadania T do i z repozytorium odbywa się zgodnie z poniższymi formułami.

$$W_{TRWrite} = f_{CRWrite}(input_T + output_T)$$

$$W_{TR\,Read} = f_{CR\,Read}(input_T + output_T)$$

gdzie $f_{CTRWrite}$ i $f_{CTRRead}$ są odpowiednio funkcjami pracy dla operacji zapisu i odczytu w repozytorium określonymi dla zasobu R komputera, na którym zlokalizowane jest repozytorium. Zasobem jest także pasmo sieciowe potrzebne do przesłania danych wejściowych lub wyjściowych pomiędzy klientem a repozytorium.
Istotną cechą funkcji pracy dla repozytorium oraz przesyłów sieciowych jest fakt, że jest ona określana dla repozytorium i nie jest zależna od danego zadania obliczeniowego. Dla wszystkich zadań uruchamianych w NetSolve, których dane wejściowe i wyjściowe umieszczane są w danym repozytorium używana jest ta sama funkcja pracy.

Z kolei model wydajności zasobów repozytorium obrazuje zależność pomiędzy stanem zasobów systemu obsługującego repozytorium a dostępnością danego zasobu repozytorium dla procesu obsługującego zapis lub odczyt danych do lub z repozytorium. W przypadku sieci model wydajności obrazuje zależność pomiędzy stanem sieci (obserwowane pasmo i opóźnienie) a dostępnością pasma dla transmisji wykonywanej przez proces przesyłający dane zadania przez sieć. W tym wypadku nie rozróżnia się operacji zapisu i odczytu. Ilość czasu potrzebną na wykonanie określonych ilości pracy związanych z zapisem i odczytem danych do i z repozytorium oraz z przesłaniem ich przez sieć oblicza się odpowiednio według następujących formuł:

$$time_{W_R Write} = g_{CR\,Rep}(W_{RWrite}, S_{Rep}),$$
$$time_{W_R Read} = g_{CR\,Rep}(W_{R\,Read}, S_{Rep}).$$

Ilość czasu potrzebną na wykonanie pracy na zasobie R repozytorium koniecznej na wykonanie zapisów i odczytów z repozytorium związanych z konkretnym zadaniem T przetwarzanym w NetSolve oblicza się według następujących formuł:

$$time_{TW_R Write} = g_{CR\,Rep}(W_{TRWrite}, S_{Rep}),$$
$$time_{TW_R Read} = g_{CR\,Rep}(W_{TR\,Read}, S_{Rep}),$$

gdzie, $W_{TRWrite}$ i W_{TRRead} są odpowiednio ilościami pracy na zasobie R koniecznymi do wykonania zapisu i odczytu do i z repozytorium danych zadania T a S_{Rep} jest zbiorem parametrów stanu zasobów repozytorium,

natomiast g_{CRrep} jest funkcją modelu wydajności zasobu R komputera, na którym zlokalizowane jest repozytorium lub funkcją modelu wydajności połączenia sieciowego pomiędzy klientem a repozytorium.
Model wydajności jest wyliczany dla danego repozytorium i jest niezależny od konkretnego zadania obliczeniowego.

Podobnie jak w przypadku funkcji pracy i funkcji modelu wydajności dla obliczeń, funkcje pracy i modelu wydajności dla repozytorium oraz przesyłów sieciowych mogą być rekonstruowane na podstawie danych pobranych podczas przetwarzania zleceń w zmodyfikowanym NetSolve dotyczących wydajności operacji na repozytorium. Mechanizm rekonstrukcji jest analogiczny do mechanizmu omówionego w pkt. 2.3.1 raportu. W fazie „uczenia" mechanizmu do oszacowania czasu zapisu i odczytu danych oraz ich przesyłania przez sieć używane są wartości domyślne określające średnią prędkość zapisu i odczytu do systemu plików repozytorium oraz średnią przepustowość dostępną w sieci pomiędzy klientem a serwerem. Mogą być one wprowadzone przez administratora systemu NetSolve do konfiguracji NetSolve lub wyliczone przez program testowy dostarczany ze zmodyfikowanym pakietem NetSolve W miarę zbierania pomiarów i uzyskania odpowiedniej liczby kompletów argumentów i wartości dla danej funkcji (domyślnie 10) funkcja pracy lub modelu wydajności jest odtwarzana i używana podczas szeregowania kolejnych wywołań funkcji dla oceny wydajności operacji na danym repozytorium.

Jeśli na węźle, na którym zlokalizowane jest repozytorium działa opcjonalny proces-monitor dla repozytorium lub repozytorium znajduje się na węźle dostępowym (wtedy działa na nim proces-monitor węzła dostępowego) wtedy dane o stanie zasobów systemu zbierane są również w sposób ciągły, niezależnie od faktu czy w danym momencie przetwarzane są zadania NetSolve. Dane pobierane ciągle są używane przez moduł predykcji czasu wykonania do oceny stanu zasobu w podczas wykonania zadania. Jeśli dane nie są pobierane ciągle, do predykcji stanu zasobu używany jest ostatni pomiar otrzymany przez system monitorowania wydajności (może on pochodzić z pomiarów wykonanych podczas ostatnio wykonywanych obliczeń).

Kwestia wydajności operacji przesyłania danych do repozytorium po stronie modułu klienta nie jest rozpatrywana przez moduł analizy. PCSS wychodzi z założenia, że z punktu widzenia szeregowania istotne są te informacje, które dotyczą repozytoriów danych związanych z różnymi serwerami NetSolve i mogą mieć wpływ na wybór konkretnego serwera dla wykonania obliczeń. Zakłada się, że liczba zasobów i wydajność operacji przesyłania i odbierania danych do i z repozytorium po stronie modułu klienta nie zależy od wyboru repozytorium.
Dlatego pełen mechanizm analizy wydajności operacji wymiany danych z repozytorium obejmujący wszystkie istotne zasoby (pamięć, dyski, sieć) po stronie klienta używany jest tylko w przypadku realizacji mechanizmu opcjonalnego wykonania lokalnego szeregowanego przez agenta. Więcej informacji na temat tego mechanizmu zawiera raport końcowy zadania WP.2.1.5 p.n. „Integracja systemu udostępniania bibliotek matematycznych ze strukturami klastra SGI". Możliwe jest natomiast uruchomienie opcjonalnego procesu monitora w module klienta, który służy jedynie do wymiany testowych paczek danych pomiędzy klientem a repozytorium. Te testowe pomiary pozwalają na ocenę pasma i opóźnienia sieci, które będzie dostępne dla przesyłu danych wejściowych i wyjściowych podczas szeregowania zadania zdalnego wykonania funkcji matematycznej wykonywanego przez agenta NetSolve.

2.5. Opracowanie metod wizualizacji danych i wizualizacji wyników analizy

Jak wspomniano w punkcie 2.2 w zmodyfikowanym *NetSolve* możliwa jest wizualizacja danych o wydajności przetwarzania w *NetSolve*. Możliwe jest automatyczne tworzenie wykresów obrazujących stan węzłów obliczeniowych i innych zasobów środowiska (połączenia sieciowe, repozytorium) i ich zmiany w czasie. Możliwe jest również obrazowanie zrekonstruowanych automatycznie funkcji pracy funkcji (więcej informacji w punkcie 2.3. raportu) oraz modelów wydajnościowych zasobów.
Wizualizacja stanu zasobów środowiska opiera się o system Orca [reference!]. Wyniki analizy wizualizowane są przy użyciu oprogramowania gnuplot [reference!]. Informacje o stanie środowiska zbierane przez proces-kolektor systemu NetSolve tworzą serie czasowe. Proces-kolektor zapisuje je w swojej bazie danych oraz umieszcza w plikach wejściowych dla oprogramowania Orca zgodnie z wymaganym przez nie formatem. Oprogramowanie Orca działa cyklicznie, niezależnie od NetSolve. Co kilka minut (domyślnie co 5) analizuje serie czasowe w plikach wypełnianych przez proces-kolektor w Agencie. W wyniku analizy Orca generuje wykresy stanu poszczególnych monitorowanych zasobów. Możliwe jest generowanie wykresów godzinnych, dziennych, tygodniowych i miesięcznych. Pliki dostępne są dla użytkowników systemu poprzez system www.
Wizualizacja wyników analizy danych o wydajności opiera się oprogramowanie gnuplot. Wykresy funkcji pracy oraz modeli wydajnościowych zasobów wykonywane są przez gnuplot na podstawie plików generowanych przez proces-analizator w Agencie NetSolve. Analizator generuje pliki dla każdej pary argument funkcji-wielkość pracy oraz dla par parametr stanu – oferowana wydajność wyrażona w procentach maksymalnej wydajności danego zasobu.
Możliwa jest również wizualizacja przebiegu wykonania poszczególnych zdalnych wywołań funkcji matematycznych w NetSolve. Ułatwia ona użytkownikom oraz administratorowi systemu analizę przebiegu przetwarzania zadań, m.in. obrazując w sposób graficzny etapy przetwarzania funkcji i ilość czasu spędzoną przez zlecenia w poszczególnych stadiach przetwarzania. Taka analiza możliwa jest dla pojedynczych wywołań funkcji lub dla wszystkich wywołań funkcji danego użytkownika lub instancji NetSolve. Wizualizacja przebiegu zadania opiera się o mechanizm dostarczany przez oprogramowanie NetLogger Visualisation (nlv) [NLV]. Oprogramowanie to umożliwia generowanie wykresów przedstawiających przebieg zadania przez poszczególne fazy przetwarzania. Wykresy tworzone są na podstawie zapisów o zdarzeniach wejścia zadania w poszczególne stadia przetwarzania zbieranych przez proces-kolektor. Zapisy o tych zdarzeniach wykonywane są przez dodatkowy kod zaszyty w kodzie oryginalnego NetSolve. Kod ten może być aktywowany odpowiednią opcją w kompilacji NetSolve (flaga –with-netlogger dodana do polecenie configure przed kompilacją). Dodatkowo logowanie tych informacji może być włączane lub wyłączane zmiennymi środowiskowymi sprawdzanymi przez system NetLogger. Wykresy umieszczane są na stronie www wraz z wykresami dotyczącymi stanu zasobów środowiska oraz wykresami zrekonstruowanych funkcji pracy i modeli wydajnościowych.

2.6. Opracowanie metod szeregowania danych

Prace w ramach zadania WP.2.1.6 zmierzały do rozszerzenia mechanizmów szeregowania dla zmodyfikowanego NetSolve tak, by dostosować go do wymagań środowiska SGIgrid oraz zwiększyć sprawność mechanizmu szeregowania. Większą sprawność szeregowania osiągnięto głównie poprzez zwiększenie dokładności oceny czasu wykonania zadań na poszczególnych zasobach.
W tym celu rozszerzono w stosunku do oryginalnego NetSolve zakres parametrów stanu środowiska obliczeniowego oraz parametrów zadań brany pod uwagę przez mechanizm predykcji czasu wykonania zdalnych zleceń. Prace te opisane są w punkcie 2.1. raportu. Z kolei opracowano architekturę systemu monitorującego wartości parametrów środowiska i zadań obliczeniowych (prace opisane w punkcie 2.2). Predykcja czasu wykonania zadań obliczeniowych w NetSolve wymaga znajomości modeli wydajnościowych zasobów środowiska. Pewne prace dotyczące istniejących modeli, które wykonano w ramach zadania WP.2.1.6. opisano w punkcie 2.3. raportu. Prace te doprowadziły do wniosku, że konieczny jest mechanizm rekonstrukcji funkcji modeli wydajności na podstawie danych o wydajności przetwarzania w danym środowisku. Opracowano metody automatycznej analizy danych dotyczących wydajności przetwarzania w NetSolve (opis prac w punkcie 2.4.), które pozwalają na wyciąganie wniosków dotyczących funkcji pracy i modelu wydajności zasobów biorących udział w przetwarzaniu. Modele te używane są przez mechanizm predykcji działający na potrzeby mechanizmu szeregowania w NetSolve (opis metody wyliczania czasu potrzebnego na przetworzenie zadania na poszczególnych węzłach obliczeniowych znajduje się również w punkcie 2.4 raportu).
Modyfikacja algorytmu szeregowania w zmodyfikowanym NetSolve polega przede wszystkim na skorzystaniu z opracowanych przez PCSS technik predykcji czasu wykonania zadania na poszczególnych zasobach. Dodatkowo, wprowadzono możliwość szeregowania umożliwiającego wyrównanie obciążeń węzłów obliczeniowych. Oryginalny mechanizm NetSolve umożliwiał tylko i wyłącznie przydział dla danego zadania

zasobu zapewniającego najkrótszy czas obsługi zadania. Wyboru trybu działania zmodyfikowanego mechanizmu szeregowania dokonuje administrator systemu za pomocą odpowiednich ustawień w konfiguracji modułu Agenta NetSolve. Domyślnie realizowana jest polityka przydziału węzła zapewniającego najwyższą wydajność.

3. Związek z pracami prowadzonymi w pozostałych zadaniach projektu

W ramach zadania WP.5 projektu celowego p.n. „Super-zarządca przewidujący czas wykonania zadania" opracowany został mechanizm zarządzania zasobami środowiska SGIgrid nazywane Brokerem klastra. Zdecydowano się nie integrować modułu szeregowania NetSolve z usługami Brokera. Za takim rozwiązaniem przemawia kilka czynników.

Po pierwsze, podczas szeregowania zadań NetSolve brane są pod uwagę specyficzne, charakterystyczne tylko dla wywołań funkcji matematycznych czynniki, m.in. informacje o wydajności poprzednich wywołań funkcji na poszczególnych zasobach obliczeniowych, szereg parametrów dotyczących zadania obliczeniowego (rozmiar zadania, wersja algorytmu, złożoność obliczeniowa, wartości parametrów modyfikujących działanie funkcji matematycznej itp.) oraz specyficzne dane dotyczące stanu systemów obliczeniowych. Dane dotyczące stanu systemów zawierają m.in. informacje o liczbie procesów w systemie, dostępnej i zajętej pamięci, obszarze wymiany (ang. *swap*), liczbie wymian na sekundę, liczbie operacji wejścia/wyjścia na sekundę w systemie itp.

Po drugie, wprowadzanie tych informacji do systemu informacyjnego Brokera jest bezcelowe. Są one pobierane z systemów obliczeniowych za pomocą specjalnych, wydzielonych, dedykowanych dla NetSolve metod. Nie są one analizowane przez Brokera na potrzeby usług innych niż system dostępu do bibliotek matematycznych. Z racji ich objętości przetwarzane są specjalnymi metodami przez Agenta NetSolve.

Po trzecie, przetwarzanie tych danych przez Broker SGIgrid nie gwarantowałoby odpowiednio krótkiego czasu odpowiedzi systemu NetSolve na zapytanie o przydział zasobów do obliczeń. W toku prac nad projektem PCSS i TASK przeprowadziło analizę i szereg testów wydajnościowych związanych z operacjami na systemie informacyjnym brokera. Z analizy i testów wynika, że wydajność tych operacji jest nieporównywalnie niższa niż operacje na lokalnej bazie danych o wydajności agenta systemu NetSolve. Dlatego całość procesów związanych z szeregowaniem zadań wykonania zdalnego funkcji matematycznych wykonuje w rozwiązaniu opracowanym przez PCSS i TASK Agent NetSolve.

Po czwarte, specyfika działania Brokera pozwala na prowadzenie szeregowania zadań wykonania funkcji matematycznych w SGIgrid w całości w module Agenta NetSolve. Informacja o zajęciu zasobów środowiska SGIgrid obliczeniami uszeregowanymi przez NetSolve trafia do Brokera za pośrednictwem systemu informacyjnego Brokera, który monitoruje stan wybranych parametrów wydajnościowych środowiska. Informacja ta trafia do Brokera z pewnym opóźnieniem. W porozumieniu z zespołem odpowiedzialnym za Broker uzgodniono jednak, że to opóźnienie jest akceptowalne, ponieważ nie wprowadza poważnych zakłóceń w proces szeregowania zadań odbywający się w Brokerze.

4. Bibliografia

[PPAM2003] M. Brzezniak, N. Meyer. "Evaluation of execution time of mathematical library functions based on historical performance information." in Parallel Processing and Applied Mathematics. R. Wyrzykowski, J. Dongarra, M. Paprzycki, J. Wasniewski. Springer-Verlag, Berlin, 2004, pp. 161-168. (Lecture Notes in Computer Science, Vol. 3019).

[RCS1] The Remote Computation System. P. Arbenz, W. Gander, and M. Oettli. Parallel Computing (23): 1421-1428, 1997.

[RSC2] The Remote Computation System. P. Arbenz, W. Gander, and M. Oettli. High Performance Computing and Networking, H. Liddell, A. Colbrook, B. Hertzberger and P. Sloot (eds.). Springer-Verlag, Berlin, 1996, pp. 820-825. (Lecture Notes in Computer Science, 1067).

[NWS1] Dynamically Forecasting Network Performance Using the Network Weather Service. Rich Wolski, Journal of Cluster Computing, Volume 1, pp. 119-132, January, 1998.

[NWS2] The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing. Rich Wolski, Neil Spring, and Jim Hayes, Journal of Future Generation Computing Systems,Volume 15, Numbers 5-6, pp. 757-768, October, 1999.

[NetLogger1] D. Gunter, B. Tierney, K. Jackson, J. Lee, M. Stoufer, "Dynamic Monitoring of High-Performance Distributed Applications" , Proceedings of the 11th IEEE Symposium on High Performance Distributed Computing, HPDC-11 , July 2002, LBNL-49698.

[NetLogger2] Strona domowa projektu NetLogger. http://www-didc.lbl.gov/NetLogger/

[NLV] Strona domowa projektu NLV. http://dsd.lbl.gov/nlv/