mgr inż. STEFAN FRYDLIŃSKI
doc. dr inż. ANDRZEJ SERWACH
Przemysłowy Instytut
Automatyki i Pomiarów MERA-PIAP
Warszawa

SYMULACJA CYFROWA DYNAMIKI PROCESÓW CIĄGŁYCH W JĘZYKU CSMP

1. Wstęp

Ostatnie dziesięciolecie w Instytucie charakteryzowało się wzrostem zapotrzebowania na badania własności dynamicznych procesów mających złożony opis matematyczny oraz rozbudowanych układów automatycznej regulacji. Znajdująca się w naszej dyspozycji maszyna analogowa ze sterowaniem logicznym WAT-1001 nie była w stanie sprostać takim zadaniom ze względu na zbyt małą pojemność operacyjną oraz trudności w realizacji dużej liczby opóźnień i złożonych zależności algebraicznych. Podjęto więc prace zmierzające do zaspokojenia potrzeb Instytutu w tym zakresie. Najmniej kosztownym rozwiązaniem było zastosowanie do tego celu symulacji cyfrowej przy wykorzystaniu maszyny cyfrowej R-32. W związku z tym zakupiono od Instytutu Systemów Sterowania System Modelowania Procesów Ciągłych (CSMP) opracowany przez firmę IBM dla maszyn cyfrowych serii 360 i przystosowany przez ISS do maszyn cyfrowych serii RIAD [1],[2],[3]. System ten został wdrożony w Laboratorium Obliczeń i Modelowania w MERA-PIAP na początku 1979 roku.

2. Podstawowe własności systemu CSMP

System Modelowania Procesów Ciągłych CSMP pracuje na maszynie cyfrowej R-32 pod kontrolą systemu operacyjnego OS/JS i zajmuje około 110 KB pamięci operacyjnej. Stanowi on pakiet programów przeznaczonych do rozwiązywania problemów opisywanych za pomocą równań różniczkowych lub analogowych schematów blokowych.
Programowanie odbywa się w skojarzonym z systemem języku CSMP opartym na Fortranie. Główną zaletą języka jest możliwość układania programu bez rozważania sekwencji instrukcji strukturalnych. Na życzenie programisty system będzie sortował automatycznie te instrukcje,zapewniając prawidłowy przepływ informacji.
System zawiera podstawowy zbiór 34 bloków funkcjonalnych. Podstawowymi blokami dynamicznymi są bloki realizujące funkcje całkowania, obliczania pochodnej oraz opóźnienia transportowego. Dwa z tych bloków — całkowania i opóźnienia transportowego — są funkcjami z pamięcią.
Jeżeli w modelu matematycznym występuje pętla algebraiczna nie zawierająca funkcji z pamięcią, to do jej rozwiązania jest przewidziany blok funkcji uwikłanej.
Do realizacji dowolnych funkcji nieliniowych przewidziano dwa generatory funkcji. Jeden z nich wykorzystuje interpolację liniową do obliczania wielkości wyjściowej przebiegu, natomiast drugi — interpolację kwadratową. Oprócz tego proste nieliniowości takie jak: idealna charakterystyka przekaźnikowa, ogranicznik amplitudy, ogranicznik nieczułości, pętla histerezy są również reprezentowane przez bloki funkcjonalne.
Do badań dynamiki większości problemów spotykanych w praktyce są niezbędne jeszcze dwa rodzaje bloków, a mianowicie bloki funkcji logicznych oraz bloki funkcji bezpośrednio zależnych od czasu.

Zestaw funkcji logicznych zawiera bloki iloczynu logicznego, negacji iloczynu, sumy logicznej,negacji sumy, inwertera, równoważności. Jest on uzupełniony blokami przerzutnika komparatora.
Funkcje bezpośrednio zależne od czasu służą głównie do tworzenia sygnałów wymuszających. Realizują je następujące bloki: funkcji skokowej, funkcji zależnej wprost proporcjonalnie od czasu, generatora impulsów oraz monoflopu. Jeżeli do tego zestawu dołączy się bloki generatorów zmiennej losowej o rozkładzie normalnym i jednorodnym oraz generatory dowolnych funkcji nieliniowych,to praktycznie każde wymuszenie, czy to standardowe, czy rzeczywiste będzie możliwe do zrealizowania.
W celu ułatwienia programowania, w systemie znajdują się jeszcze cztery bloki zbudowane z bloków podstawowych przy wykorzystaniu podprogramów systemowych typu MACRO. Są to::

- integrator sterowany; blok ten zależnie od sygnałów sterujących albo całkuje, albo pamięta ostatnią wartość całki, albo śledzi warunek początkowy; jest on bardzo przydatny na przykład przy symulacji regulatorów, w których stan pracy całki zależy od położenia elementu wykonawczego;
- inercja pierwszego rzędu;
- inercja pierwszego rzędu, w której wielkość wejściowa zależy od funkcji i jej pochodnej;
- inercja drugiego rzędu.

Podstawowy zbiór bloków funkcjonalnych jest uzupełniony biblioteką Fortranu i być może zostanie w przyszłości uzupełniony biblioteką funkcji matematycznych.
Użytkownik dysponując wymienionym zbiorem funkcji CSMP i Fortranu może tworzyć własne,większe bloki funkcjonalne. Są możliwe cztery typy bloków funkcjonalnych definiowanych przez użytkownika:

- Funkcja MACRO — raz zdefiniowana funkcja MACRO może być wielokrotnie używana wewnątrz symulacyjnych instrukcji strukturalnych w programie głównym. Wewnątrz MACRO mogą być użyte dowolne instrukcje strukturalne CSMP i Fortranu. Nie mogą tam występować instrukcje danych i instrukcje sterujące CSMP. Nie mogą tam również występować Fortranowskie instrukcje sterujące i instrukcje wejścia/wyjścia, chyba, że są zawarte wewnątrz funkcji PROCEDURE. Instrukcje wewnątrz MACRO są automatycznie sortowane.
- Funkcja PROCEDURE — instrukcje wewnątrz PROCEDURE nie są sortowane, zatem należy zwracać baczną uwagę na właściwą ich kolejność. Można tu umieszczać dowolne instrukcje strukturalne CSMP i Fortranu.Definiowanie funkcji PROCEDURE daje poważne korzyści ze względu na możliwości stosowania zmiennych logicznych, zmiennych ze wskaźnikami i skoków. Funkcja PROCEDURE jest pojedynczym blokiem funkcjonalnym, chociaż definicja jej wymaga kilku instrukcji.
- Funkcja Fortranowska — wewnątrz FUNCTION można wykorzystywać wszelkie możliwości Fortranu oraz stosować dowolne standardowe bloki funkcjonalne CSMP za wyjątkiem funkcji z pamięcią, historią, funkcji uwikłanej oraz funkcji zdefiniowanych za pomocą MACRO systemowego i użytkownika.
- Subrutyna Fortranowska (SUBROUTINE) — jest stosowana w przypadku konieczności generowania przez blok funkcjonalny wielu wyjść. Własności i ograniczenia są analogiczne jak w przypadku FUNCTION.

Podprogramy Fortranowskie mogą być zakatalogowane do biblioteki CSMP przez załadowanie ich z biblioteką Fortranu za pomocą odpowiedniej procedury systemu operacyjnego.
Możliwości tworzenia własnych bloków funkcjonalnych mogą być wykorzystane na przykład do opracowania własnej metody całkowania, jeżeli, zdaniem użytkownika, żadna z siedmiu dostarczonych przez CSMP metod nie spełni jego wymagań. Wśród metod standardowych dwie są metodami zmiennokrokowymi, a pięć — stałokrokowymi.
Na zakończenie opisu własności systemu należy wspomnieć o budowie programu oraz o sterowaniu przebiegiem obliczeń.
Program w języku CSMP może być złożony z trzech segmentów: początkowego, dynamicznego i koń-

cowego. Jedynie segment dynamiczny jest niezbędny w programie, pozostałe są opcjonalne.
Segment początkowy jest przeznaczony do obliczania warunków początkowych i parametrów w zależności od parametrów bardziej podstawowych. Obliczenia w tym segmencie są przeprowadzane jeden raz na początku obliczeń.
Segment dynamiczny zawiera kompletny opis dynamiki układu oraz obliczeń wykonywanych podczas każdego przebiegu.
Segment końcowy jest używany do obliczeń wymaganych po zakończeniu każdego przebiegu. Mogą to być nieskomplikowane korekty parametrów lub na przykład dobór parametrów do spełnienia funkcji celu według założonego algorytmu optymalizacyjnego. Należy jeszcze dodać, że każdy z segmentów może być, zgodnie z życzeniem programisty, dzielony na sekcje sortowane i niesortowane.
System CSMP pozwala użytkownikowi sterować procesem obliczeń zgodnie ze specyficznymi wymaganiami problemu. Jeżeli jest wymagana odpowiedź układu dla jednego zestawu parametrów, wówczas obliczenia są przeprowadzane jednokrotnie dla złożonego czasu przebiegu (FINTIM) lub do czasu, w którym zostaną spełnione wyspecyfikowane w FINISH warunki zakończenia liczenia. Podczas jednokrotnego rozwiązywania zadania można zatrzymać liczenie, zmienić wartości określonych parametrów fizycznych czy systemowych lub metodę całkowania i kontynuować rozwiązywanie zadania w nowych warunkach. Chcąc znać odpowiedzi układu dla kilku zestawów parametrów, otrzymamy je automatycznie, jeżeli w odpowiedni sposób zadeklarujemy te zestawy w segmencie końcowym programu. Jest również możliwe liczenie iteracyjne. Wówczas w segmencie końcowym następuje obliczenie funkcji celu, porównanie jej z wartością zadaną i, o ile warunki porównania nie zostały spełnione, skorygowanie wytypowanych parametrów oraz powtórzenie przebiegu dla nowych parametrów. Zakończenie obliczeń następuje po spełnieniu warunków porównania.
Wyniki symulacji mogą być wyprowadzane w postaci wydruku i (lub) wykreślenia zmiennych na drukarce wierszowej. Dodatkowo można zażądać wydrukowania minimalnych i maksymalnych wartości wytypowanych zmiennych oraz zapisania zbioru danych bądź na taśmie, bądź na dysku w celu zapisania ich w późniejszym okresie na rejestratorze cyfrowym. Ostatni sposób nie jest praktykowany z powodu braku w Laboratorium Obliczeń i Modelowania odpowiedniego rejestratora cyfrowego. Prócz tego są dostępne Fortranowskie możliwości wyprowadzania wyników obliczeń.

3. Ważniejsze ograniczenia programowe

Ze względu na ograniczony obszar pamięci zarezerwowany dla systemu istnieją ograniczenia dotyczące pewnych składowych. Ważniejsze z nich są wyszczególnione niżej:

- Liczba nazw zmiennych wyjściowych łącznie z nazwami wyjść pośrednich generowanych przez CSMP dla funkcji MACRO i INTGRL (całka) $\leqslant 500$
- Liczba nazw zmiennych wejściowych łącznie z nazwami parametrów $\leqslant 1400$
- Liczba nazw parametrów $\leqslant 400$
- Liczba integratorów + funkcje z pamięcią + funkcje z historią $\leqslant 300$
- Liczba funkcji z pamięcią i historią dostarczanych przez użytkownika $\leqslant 50$
- Liczba funkcji z pamięcią dostarczonych przez użytkownika $\leqslant 15$
- Liczba tablic $\leqslant 25$
- Liczba instrukcji strukturalnych w pojedynczej sekcji SORT łącznie z wygenerowanymi przez CSMP dla funkcji MACRO i INTGRL $\leqslant 600$
- Liczba drukowanych zmiennych wyjściowych łącznie z czasem $\leqslant 50$
- Liczba drukowanych i wykreślanych zmiennych wyjściowych łącznie z czasem $\leqslant 50$
- Liczba zmiennych wyjściowych przygotowanych do zapisu na rejestratorze cyfrowym łącznie z czasem $\leqslant 50$

- Liczba warunków zakończenia przebiegu (FINISH) $\leqslant 10$
- Liczba nazw zmiennych stałoprzecinkowych $\leqslant 20$

4. Uwagi końcowe

W okresie rocznego użytkowania systemu [4] zdobyto pierwsze doświadczenia dotyczące przydatności języka CSMP do symulacji cyfrowej procesów ciągłych oraz wad i zalet tego systemu w porównaniu z symulacją na maszynach analogowych ze sterowaniem logicznym. Poniżej podano takie porównanie.

Budowa modelu matematycznego, który ma być symulowany z użyciem CSMP, wymaga znacznie większej uwagi i sumienności niż budowa modeli symulowanych na maszynie analogowej. Wynika to stąd, że błędy modelu są bardzo łatwe do wykrycia przy symulacji analogowej z uwagi na, z natury rzeczy, konwersacyjny charakter pracy z taką maszyną. Błędy w modelu symulowanym w CSMP wymagają żmudnej analizy, bowiem interpretacja wyników przebiegów dynamicznych podawanych w formie drukowanych tabel lub pojedynczych wykresów nie jest przystępna, zwłaszcza dla konstruktorów przyzwyczajonych do posługiwania się zbiorami wykresów.

Najmniej kłopotów sprawia symulacja w CSMP modeli opisywanych równaniami różniczkowymi liniowymi (transmitancjami), ale z praktycznego punktu widzenia, czyli w przypadku komputerowo wspomaganego projektowania są to problemy banalne. Dlatego nie należy do tej zalety przywiązywać dużej wagi.

Znaczną wygodą dla programisty problemu w CSMP jest brak konieczności skalowania zmiennych, co zawsze było uciążliwe w przypadku maszyn analogowych. Jednakże ta wygoda kryje w sobie niebezpieczeństwo posługiwania się modelami o bardzo rozrzuconych wartościach parametrów dynamicznych (stałych czasowych); należy wówczas model uprościć i pominąć wpływy dynamiczne nie mające znaczenia. Wykrycie tej nieprawidłowości w programie analogowym było sprawą elementarną-na niewłaściwą sytuację wskazywały duże różnice wartości współczynników przy integratorach. W CSMP można czasem tę nieprawidłowość wykryć dopiero wówczas, gdy maszyna zasygnalizuje przepełnienie, co oznacza już bardzo duży błąd w modelu.

Niewątpliwą zaletą jest możliwość łatwego symulowania złożonych, nieliniowych zależności algebraicznych i opóźnień transportowych. Pod tym względem maszyny analogowe ustępują całkowicie systemowi CSMP.

Odpowiednikiem znanego w maszynach analogowych zjawiska wzmacniania się szumów własnych maszyny przy symulacji dużych i zwłaszcza silnie nieliniowych modeli z dużą liczbą elementów statycznych jest w CSMP rozbieganie się przebiegów obliczanych, spowodowane kumulacją błędów liczenia z kroku na krok całkowania. W obu przypadkach skuteczną ochroną jest wprowadzenie do modelu filtrów dolnoprzepustowych, ale, aby filtry te nie zakłócały przebiegów, jest konieczne w obu przypadkach wydłużanie czasu obliczeń. W CSMP jest to bezpośrednim skutkiem konieczności zagęszczenia kroku całkowania. Czasem wystarcza samo zagęszczenie kroku całkowania bez filtracji sygnałów. W każdym razie, maszyna prowadzi liczenie w cyklu granicznym, który ma częstotliwość porównywalną z krokiem całkowania, a amplitudę zależną od rozbudowania problemu symulowanego i dobranego kroku całkowania. To zjawisko powoduje, że wbrew rozpowszechnionemu przekonaniu o wyjątkowo wysokiej dokładności maszyn cyfrowych, dokładność rozwiązywania złożonych problemów dynamicznych w CSMP jest porównywalna z dokładnością, jaką można uzyskać przy symulacji analogowej.

Przy symulacji złożonych problemów czas liczenia (pojedynczego przebiegu) układu równań rośnie z dość wysoką potęgą w stosunku do wzrostu liczby instrukcji strukturalnych zwłaszcza reprezentujących funkcjonalne bloki dynamiczne CSMP. Jest to spowodowane nie tylko samym rozbudowa

niem problemu lecz również związaną z tym koniecznością skracania kroku całkowania, co wyjaśniono wyżej. W rezultacie, przy złożonych problemach stosowanie automatycznej optymalizacji jest raczej niemożliwe, bowiem w takim przypadku do obliczania optymalnej wartości wskaźnika jakości jest konieczne zazwyczaj powtórzenie kilkadziesiąt do kilkuset razy (zależnie od stosowanej metody) rozwiązania równań opisujących model. Z praktycznego punktu widzenia jest to w gruncie rzeczy ograniczenie mało istotne. Rzadko kiedy konstruktor mógłby użyć klasycznej optymalizacji ciągłej ze względu na to, że ograniczenia normalizacyjne, dostępność handlowa elementów układu, konieczność unifikacji itp. sprowadzają zwykle problem wyboru do rozpatrzenia kilkunastu wariantów, co łatwiej zrobić analizując kilkanaście normalnych rozwiązań i wyboru dokonać według uznania. Nawet w przypadku parametrów, które można dobierać płynnie, wystarczające jest zazwyczaj zbadanie zachowania się układu dla kilku wartości w możliwym do zrealizowania przedziale zmienności danego parametru.

Literatura

[1] CSMP. System modelowania procesów ciągłych na emc JS. Opis języka. Instytut Systemów Sterowania. Katowice.

[2] CSMP. System modelowania procesów ciągłych na emc JS. Podręcznik programisty systemowego, tom 1. Instytut Systemów Sterowania. Katowice.

[3] CSMP. System modelowania procesów ciągłych na emc JS. Podręcznik programisty systemowego, tom 2. Instytut Systemów Sterowania. Katowice.

[4] Rozwój oprogramowania dla badań sprzętu systemu POLMATIK. Sprawozdanie MERA-PIAP nr rej. 2720, 1979.