SCAPROC als CPU- und Speicherfresser

Luzia Tinten
viadee Unternehmensberatung AG
Anton-Bruchhausen-Straße 8
48147 Münster
luzia.tinten@viadee.de

Daniel Schulte
viadee Unternehmensberatung AG
Anton-Bruchhausen-Straße 8
48147 Münster
daniel.schulte@viadee.de

Zusammenfassung

Die SAS-Prozedur SCAPROC kann ein nützliches Werkzeug sein, um Programmverläufe und Datennutzung zu protokollieren und zu analysieren. Es gibt jedoch Bedingungen, unter welchen Jobs damit an CPU- und Speichergrenzen stoßen. So gibt es Faktoren, welche die Prozedur zum Speicherfresser machen. Daraus ergibt sich wiederum, wann SCAPROC sinnvoll und nützlich ist und wann auf den Einsatz der Prozedur verzichtet werden sollte.

Schlüsselwörter: PROC SCAPROC, INITSTMT, FULLSTIMER, Memory, Real Time, User CPU Time, ATTR, OPENTIMES, INTCON, EXPANDMACROS

1 Einleitung

In vielen Unternehmen findet die Base SAS Software Anwendung bei der Datenverarbeitung. Projekte, welche Base SAS verwenden, bestehen in der Regel aus mehreren SAS-Programmen. Dabei dienen Ausgaben eines Programmes häufig als Eingaben anderer Programme. Kommt es zu Veränderungen, ziehen sich diese schnell durch mehrere Programme und haben Einfluss auf eine Reihe von SAS-Jobs. Da sich bei einer größeren Anzahl von Programmen komplexe Abhängigkeiten ergeben können, ist die Wartung und Pflege entsprechend aufwändig.

Die SAS Software bietet mit der Prozedur SCAPROC hier ein nützliches Hilfsmittel. Die Prozedur erlaubt es dem Programmierer, eine zusätzliche Log-Datei zu erzeugen, welche für jeden Programm-Schritt die Ein- und Ausgaben übersichtlich protokolliert. Abhängigkeiten werden sichtbar und Datenflüsse nachvollziehbar. Dies erleichtert die Analyse von Programm-verläufen immens. Allerdings scheint SCAPROC nicht performant zu arbeiten, da die Prozedur unter bestimmten Umständen einen sehr hohen CPU- und Speicherbedarf verursacht, welcher schlimmstenfalls zu Programmabbrüchen führt. Dies ist besonders kritisch, da die entsprechenden Fehlermeldungen (s.u.) nicht offensichtlich mit der Prozedur SCAPROC in Verbindung gebracht werden. Das Problem lässt sich vermeiden, indem die Prozedur nicht standardmäßig mitläuft, sondern gezielt und bedarfsorientiert eingesetzt wird. Dieser Beitrag analysiert den Einfluss der Prozedur SCAPROC auf Performance und Speicher-Ressourcen. Zudem wird diskutiert, wann der Einsatz von SCAPROC empfehlenswert ist.

2 PROC SCAPROC

Seit der Version 9.2 ist die Prozedur SCAPROC Bestandteil der Base SAS Software. Die Prozedur implementiert den SAS Code Analyzer. Bei einem Aufruf der Prozedur werden alle nachfolgenden I/O Informationen in eine festgelegte Log-Datei geschrieben. Die Syntax sieht folgendermaßen aus: Zu Programmbeginn wird mit

```
PROC SCAPROC;
   RECORD 'TEST.txt';
RUN;
```

die Protokollierung initiiert. Im RECORD-Aufruf wird festgelegt, in welche Datei das SCAPROC-Log geschrieben werden soll. Mittels

```
PROC SCAPROC;
   WRITE;
RUN;
```

wird die Protokollierung beendet und die zwischenzeitlich gesammelten Informationen werden in die bei der Initialisierung definierte Datei geschrieben. Existiert der zweite Programmaufruf nicht, so werden alle Informationen erst zum Ende der SAS-Sitzung in die Log-Datei geschrieben.

Durch Angabe von `ATTR`, `OPENTIMES`, `INTCON` können der Log-Datei weitere Informationen hinzugefügt werden. So schreibt `ATTR` Informationen über die Variablen des INPUT-Datasets ins Log. `OPENTIMES` schreibt Zugriffsinformationen in das Log und `INTCON` schreibt die Integritätsbedingungen ins Log. Die Anweisung `EXPANDMACROS` löst Makros in der Protokollierung auf.

```
PROC SCAPROC;
   RECORD 'TEST.txt' ATTR OPENTIMES INTCON EXPANDMACROS;
RUN;
```

SCAPROC kann prinzipiell an einem beliebigen Punkt in ein SAS Programm integriert werden, eleganter ist es jedoch, die Prozedur per INITSTMT und TERMSTMT in die SAS-Konfiguration zu übernehmen. Anweisungen, die mit INITSTMT erfolgen, werden nach Anweisungen in einer AUTOEXEC-Datei und vor Anweisungen in der SYSIN Datei ausgeführt. Anweisungen mit TERMSTMT werden nach Ende der SAS-Sitzung ausgeführt. Das folgende Beispiel zeigt die entsprechende Syntax.

```
"[Pfad SAS.exe]" -initstmt "proc scaproc; record [Log-Datei]; run;"
-termstmt "proc scaproc; write; run;" -sysin "[SAS-Programm]"
```

So wird zu jedem SAS-Aufruf die Protokollierung durch SCAPROC durchgeführt.
Die mit SCAPROC erzeugte Log-Datei hat dann zum Beispiel folgenden Aufbau [1]:

```
/* JOBSPLIT: DATASET OUTPUT SEQ WORK.A.DATA */
/* JOBSPLIT: LIBNAME WORK ENGINE V9 PHYS
C:\DOCUME~1\userid\LOCALS~1\Temp\SAS Temporary Files\_TD1252 */
/* JOBSPLIT: ELAPSED 3984  */
/* JOBSPLIT: PROCNAME DATASTEP */
/* JOBSPLIT: STEP SOURCE FOLLOWS */

data a;
    do i = 1 to 1000000;
        j = cos(i);
        output;
    end;
run;

/* JOBSPLIT: END */
```

Das SCAPROC-Log ist ein ausführbares SAS Programm, bestehend aus dem Programmcode, welcher ausgeführt wurde, sowie JOBSPLIT Kommentaren, welche die eigentlichen Log-Informationen enthalten. Jede Ausführung einer SAS Prozedur oder eines DATA Steps ist im Log ein Task. Zu jedem Task werden I/O Informationen gesammelt und im Log pro Task geordnet ausgegeben. Entsprechende Schlüsselwörter wie `INPUT`, `OUTPUT`, `UPDATE` spezifizieren die jeweilige Aktivität. Alle Schlüsselwörter sind in der SAS Dokumentation zur Prozedur SCAPROC aufgeführt und erläutert [1]. Die Log-Datei kann in einem nächsten Schritt noch nachbearbeitet und anderweitig abgelegt werden, was aufgrund der Schlüsselwörter unkompliziert umsetzbar ist.

3 Beispiel

Im Folgenden geht es um die Auswirkung der Prozedur SCAPROC auf Ressourcen wie Speicher, CPU und Gesamtlaufzeit und somit auf die Performance. Wird in Programmen viel mit SAS-Makros gearbeitet, kommt es durch SCAPROC zu einem signifikant höheren CPU- und Speicherbedarf und einer dramatischen Verlängerung der Bearbeitungszeit. Zudem wirken sich die Zusatzoptionen der Prozedur (ATTR, OPENTIMES, INTCON und EXPANDMACROS) unterschiedlich auf die Performance aus. Dieses Verhalten kann an einem einfachen Beispiel analysiert werden.

```
%macro test(loop);
  %do i=1 %to &loop.;
    data test&i.;;
      set sashelp.cars;
    run;
  %end;
%mend test;
%test(10);
```

Das Programm enthält einen Makro-Aufruf. Hier lässt sich die Programm-Laufzeit durch die Schleifenzahl (engl. loop) variieren. Somit ist eine Performance-Analyse in Abhängigkeit der Schleifenzahl möglich. Gemessen werden die unterschiedlichen Ressourcen mithilfe der System Option FULLSTIMER. FULLSTIMER liefert Informationen über die Performance einer SAS Prozedur oder eines DATA Steps und über die der gesamten SAS Sitzung. Die Option liefert verschiedene Parameter zur Performance und zum Ressourcenbedarf. Die Option ist dafür geeignet, Performance Probleme im Programmverlauf zu lokalisieren und Programme hinsichtlich der Laufzeit und des Ressourcenbedarfs zu optimieren. Die Informationen werden im SAS-Log ausgegeben. Die Ausgabe im Log sieht dann folgendermaßen aus:

```
PROZEDUR MEANS used (Total process time):
real time               0.02 seconds
user cpu time           0.03 seconds
system cpu time         0.00 seconds
memory                  7509.87k
OS Memory               36020.00k
Timestamp               18.02.2019 11:49:39 vorm.
Step Count                          21  Switch Count  1
Page Faults                         0
Page Reclaims                       2070
Page Swaps                          0
Voluntary Context Switches          49
Involuntary Context Switches        2
Block Input Operations              0
Block Output Operations             0
```

Die Bedeutung der unterschiedlichen Parameter ist in der SAS Dokumentation entsprechend erläutert [2]. Hier werden nur einige der Parameter betrachtet.

Real Time/ User CPU Time: Real Time bezeichnet die tatsächlich verstrichene Zeit (engl. elapsed time), die ein Step oder ein gesamter Job von Anfang bis Ende benötigt. User CPU Time bezeichnet die Zeit, die der Prozessor zur Ausführung des Usergeschriebenen Programmcodes benötigt.

Memory/ OS Memory: Memory bezeichnet die Menge an Speicher die benötigt wird, um den Step oder den gesamten Job auszuführen. OS Memory bezeichnet die Menge an Speicher, die SAS während der Ausführung des Steps oder des gesamten Jobs maximal zur Verfügung steht. Dies beinhaltet auch Speicher, den das System für SAS allokiert, welcher aber vom SAS Prozess nicht in Anspruch genommen wird.

Nicht bekannt sind Messgenauigkeit und Fehler der einzelnen Messgrößen. Zudem sind Ressourcenkapazitäten abhängig von der verwendeten Hardware und der Ressourcenbedarf ist für verschiedene Programme unterschiedlich. Somit geht es hier lediglich um eine qualitative Analyse, wie sich SCAPROC auf die Performance auswirkt. Die konkreten Zahlenwerte sind nicht relevant.

Das Beispiel-Programm zur Analyse wird zunächst einmal mit und einmal ohne die Prozedur SCAPROC in Abhängigkeit der Schleifenzahl ausgeführt. Die folgenden Abbildungen zeigen den Verlauf der CPU-Zeit und des Speicherbedarfs in Abhängigkeit der Schleifenzahl.

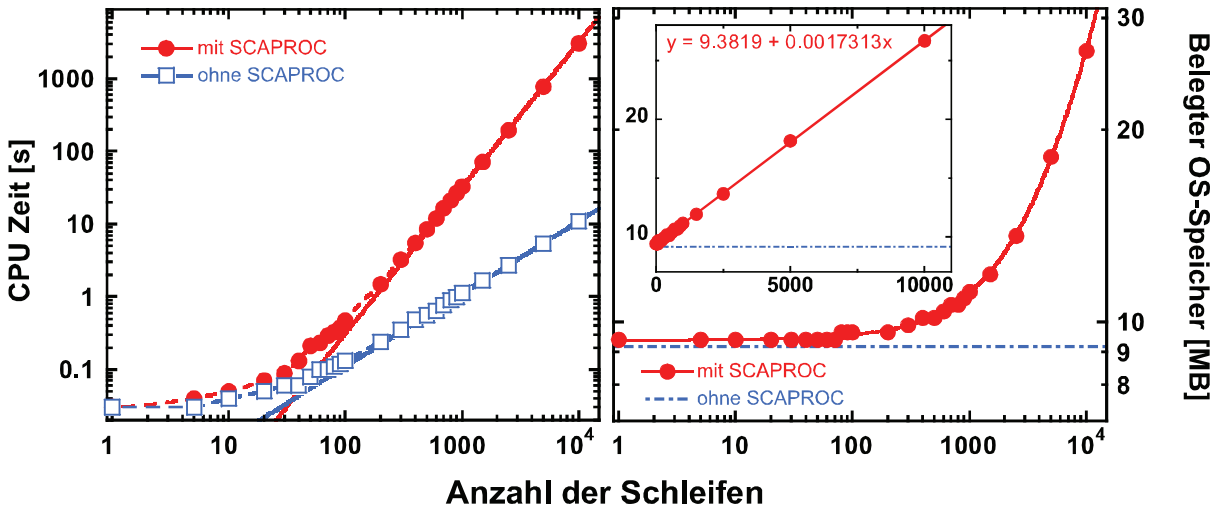

Abbildung 1: User CPU Time (links) und OS Memory (rechts) in Abhängigkeit der Schleifenzahl. User CPU Time ist doppelt logarithmisch dargestellt.

Abbildung 1 zeigt die CPU-Zeit (User CPU Time) und den Speicherbedarf (OS Memory) in Abhängigkeit der Schleifenzahl. Die Auftragung der CPU-Zeit in Abhängigkeit der Schleifenzahl ist doppelt logarithmisch. Für sehr kleine Schleifenzahlen unterscheidet sich die Laufzeit bei Ausführung mit und ohne SCAPROC kaum oder nur wenig. Erst mit steigender Schleifenzahl unterscheiden sich die Ausführungsvarianten hinsichtlich der Laufzeit. Der Verlauf lässt sich in der doppelt logarithmischen Darstellung jeweils mit einer Geraden (ohne SCAPROC Steigung = 1, mit SCAPROC Steigung = 2) beschreiben. Während bei der Ausführung ohne SCAPROC die Laufzeit also linear mit der Schleifenzahl zunimmt, ist die Zunahme mit SCAPROC quadratisch mit der Schleifenzahl. Je größer die Schleifenzahl desto gravierender ist demnach die Zunahme in der Laufzeit, wenn die Prozedur SCAPROC mitläuft.

Der Speicherbedarf ist in der Ausführung ohne SCAPROC unabhängig von der Schleifenzahl konstant. Mit SCAPROC nimmt der Speicherbedarf mit steigender Schleifenzahl zu. Die Zunahme ist hier linear und unabhängig von der Größe der im Testprogramm verwendeten Eingabedatei.

Im Weiteren werden auch die Auswirkungen der SCAPROC-Zusatzoptionen in unterschiedlichen Varianten auf die Performance untersucht. So können die Optionen alle eingeschaltet werden, oder aber auch einzeln, bzw. in beliebiger Kombination. Verglichen wird hier mit der Ausführung ohne SCAPROC sowie der Ausführung mit SCAPROC aber ohne Zusatzoptionen.

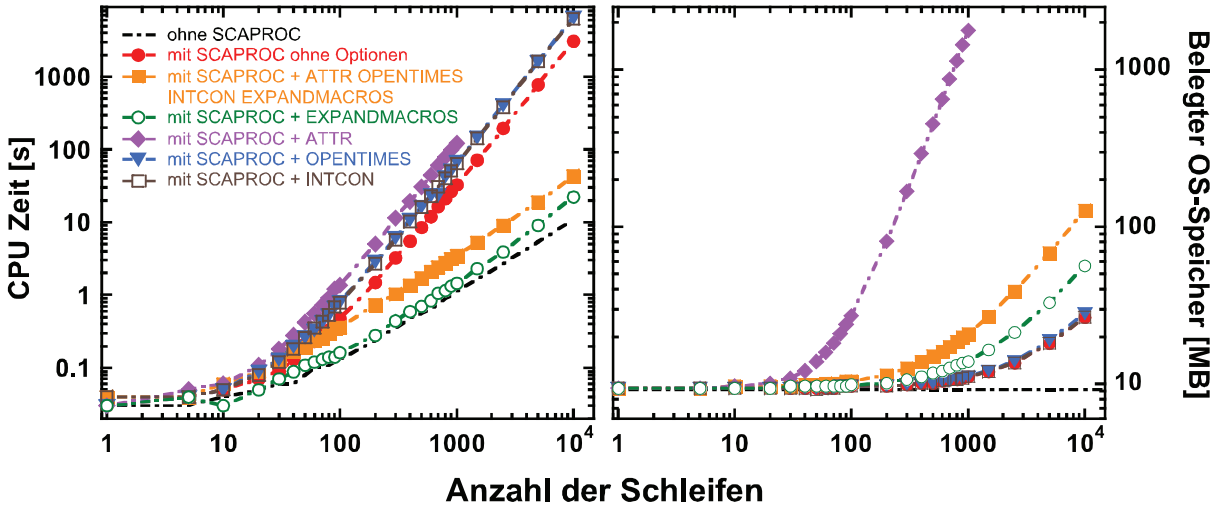

Abbildung 2: User CPU Time (links) und OS Memory (rechts) in Abhängigkeit der Schleifenzahl für verschiedene SCAPROC Zusatzoptionen. User CPU Time ist doppelt logarithmisch dargestellt.

Abbildung 2 zeigt die CPU-Zeit sowie den Speicherbedarf in Abhängigkeit der Schleifenzahl für die unterschiedlichen Optionen. Es ist zu erkennen, dass sich die verschiedenen Optionen sehr unterschiedlich auf die Laufzeit und den Speicherbedarf auswirken. Während die Option EXPANDMACROS z.B. CPU-technisch performanter ist als die Variante mit SCAPROC ohne Zusatzoptionen, so ist der Speicherbedarf mit EXPANDMACROS höher als in der Variante ohne Zusatzoptionen.

Besonders auffällig wirkt sich im gewählten Beispiel-Programm die Option ATTR auf die Performance aus. So ist hier die Laufzeit von allen Ausführungsvarianten am höchsten, und der Speicherbedarf nimmt quadratisch mit der Schleifenzahl zu, während die Zunahme bei den restlichen Varianten linear ist. Zudem läuft das Programm aufgrund des hohen Speicherbedarfs bei hoher Schleifenzahl (hier >1075) nicht mehr vollständig durch und es wird folgende Fehlermeldung ausgegeben, welche jedoch keine Rückschlüsse darauf zulässt, dass der Abbruch mit SCAPROC im Zusammenhang steht:

```
FATAL: Insufficient memory to execute DATA step program. Aborted
during the phase.
ERROR: The SAS System stopped processing this step because of
insufficient memory.
```

Interessant ist jedoch, dass dieses Problem in der Ausführungsvariante mit SCAPROC und allen Optionen (also auch ATTR) nicht auftaucht. Es scheint, dass die Optionen kombiniert einen anderen Einfluss auf die Performance haben als einzeln, und dass sich bestimmte Optionen neutralisieren hinsichtlich des CPU- und Speicherbedarfs. Diese Beobachtung ist im Folgenden näher betrachtet. Da es im gewählten Beispiel vor allem die Option ATTR ist, welche bei alleiniger Verwendung zu einem extrem hohen Speicherbedarf führt, gemeinsam mit den anderen Optionen jedoch einen weit geringeren Speicherbedarf verursacht, wird diese Option nun jeweils in Kombination mit einer der drei anderen Optionen betrachtet.

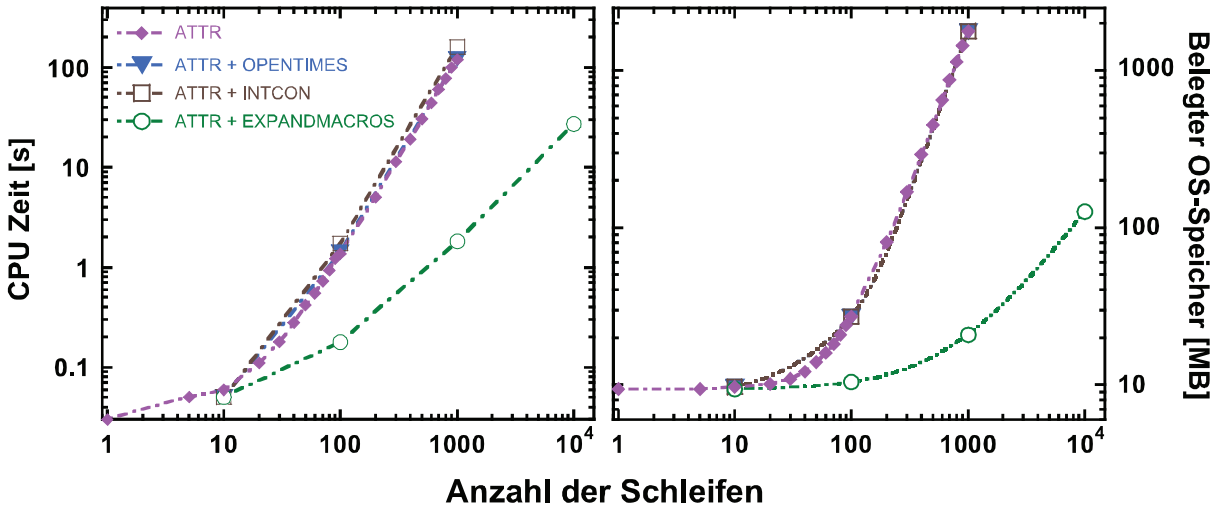

Abbildung 3: User CPU Time (links) und OS Memory (rechts) in Abhängigkeit der Schleifenzahl für die SCAPROC Zusatzoption ATTR – alleine und in Kombination mit anderen Optionen. User CPU Time ist doppelt logarithmisch dargestellt.

Abbildung 3 zeigt CPU-Zeit und Speicherbedarf in Abhängigkeit der Schleifenzahl für SCAPROC mit der Zusatzoption ATTR sowie für SCAPROC mit der Option ATTR und jeweils einer der drei weiteren Optionen. Zu erkennen ist, dass sich OPTENTIMES und INTCON kaum auf die schlechte, durch ATTR verursachte Performance auswirken. So kommt es in beiden Fällen, wie in der Ausführungsvariante mit SCAPROC und nur ATTR, zu einem Fehler aufgrund von zu hohem Speicherbedarf. Die Option EXPANDMACROS jedoch wirkt sich positiv aus, und verbessert die Performance entscheidend. Offensichtlich ist auch in der Ausführungsvariante mit allen Zusatzoptionen EXPANDMACROS dafür verantwortlich, dass die Performance gegenüber der Ausführungsvariante mit SCAPROC und ohne Zusatzoptionen durch ATTR nicht massiv verschlechtert wird.

Insgesamt lässt sich anhand des Beispiels zeigen, dass sowohl CPU- als auch Speicherbedarf signifikant zunehmen, wenn die Prozedur SCAPROC eingebunden ist. Die Zusatzoptionen der Prozedur wirken sich unterschiedlich aus und können Performance und Ressourcenbedarf einerseits verbessern, andererseits aber auch massiv verschlechtern. Aufgrund des gesteigerten Speicherbedarfs kommt es zu einer Auslastung des realen Arbeitsspeichers mit der Folge, dass langsamer SWAP Space verwendet wird. Schlimmstenfalls kommt es zum Job-Abbruch in Folge eines Speicher-Fehlers. Der gesteigerte CPU-Bedarf bewirkt insgesamt längere Laufzeiten und eine deutlich erhöhte Last.

5 Diskussion

Schon ein simples Beispiel zeigt die Probleme auf, die gegebenenfalls auftreten, wenn die Prozedur SCAPROC verwendet wird. Irreführend ist hier, dass die Fehler und Performance-Probleme, die auftauchen, nicht direkt und eindeutig mit dieser Prozedur in Verbindung gebracht werden können.

Der qualitative Unterschied im Speicherbedarf mit und ohne SCAPROC deutet darauf hin, dass die Performance-Probleme der Prozedur mit zurückgehaltenem Speicher zusammenhängen. Bei der Ausführungsvariante ohne SCAPROC ist der Speicherbedarf auch bei steigender Schleifenzahl konstant. Offensichtlich wird der Speicher nach jedem Schleifenaufruf freigegeben. Bei der Ausführungsvariante mit SCAPROC steigt der Speicherbedarf mit steigender Schleifenzahl. Speicher wird in irgendeiner Form nicht freigeben.

Nichtsdestotrotz stellt die Prozedur SCAPROC ein nützliches Hilfsmittel bei der Analyse von Programmverläufen dar, weshalb auf den Einsatz nicht verzichtet werden sollte. In Entwicklungs- und Testumgebungen liefert die Prozedur nützliche Daten. Die Zusatzoptionen der Prozedur sind hilfreich, sollten jedoch nicht uneingeschränkt zugeschaltet werden, sondern lediglich dann, wenn die entsprechende Log-Information tatsächlich benötigt wird. In produktiven Umgebungen sollte die Routine nicht standardmäßig implementiert werden, da Performance und Ressourcenbedarf signifikant beeinflusst werden und die daraus resultierenden Probleme plötzlich und unerwartet auftreten können.

Literatur

[1] Base SAS 9.2 Procedures Guide, PROC SCAPROC, http://support.sas.com/documentation/cdl/en/proc/61895/HTML/default/viewer.htm#a003199742.htm

[2] Base SAS Support, Scalability and Performance, FULLSTIMER SAS Option https://support.sas.com/rnd/scalability/tools/fullstim/index.html