独立行政法人 情報通信研究機構

Para-SimString システム マニュアル

《一般公開版》

作成者：若松 隆司

作成日：2012/02/24
更新日：2013/03/28

はじめに

Para-SimString システムは大量のテキストデータの中から指定された文字列に類似する一行を探し出す「並列」類似文字列検索システムです。検索エンジン部分には岡崎直観氏の類似文字列検索ライブラリ SimString を利用しています。Para-SimString はクラスタ環境での分散処理により、大量のテキストデータを高速に処理することが可能です。コマンドラインでの検索スクリプトの実行による検索と、Web インターフェースによる検索の両機能を備えています。

このマニュアルは Para-SimString システムの一般公開版について記したものです。初めにシステム構築にあたって考えておくべきことについての記載があります。Para-SimString システムがどの様な構成要素から成り立っているのか、どの様なアーキテクチャとなっているのか、検索対象データはどの様な形で保持されるのか、といった事前に決定すべき問題について基本的な指針を示しています。

次に具体的なシステムのインストールや設定の方法が記載されています。システムを構成するのに必要なサーバのスペック、ソフトウェア、環境、設定についての説明があり、その後に構築するサーバの種類毎にシステムのインストール手順を説明しています。

最後にシステムの実際の使用方法について記しています。検索を行う前に作成しなければならないインデックスの作成方法、コマンドラインで検索を行う方法、Web インターフェースを使って検索を行う方法が記載されています。

目次

構築計画

1 システム概要

ここでは Para-SimString システムの概要を記します。まず、以下に Para-SimString システムを使用する際のイメージ図を記します。

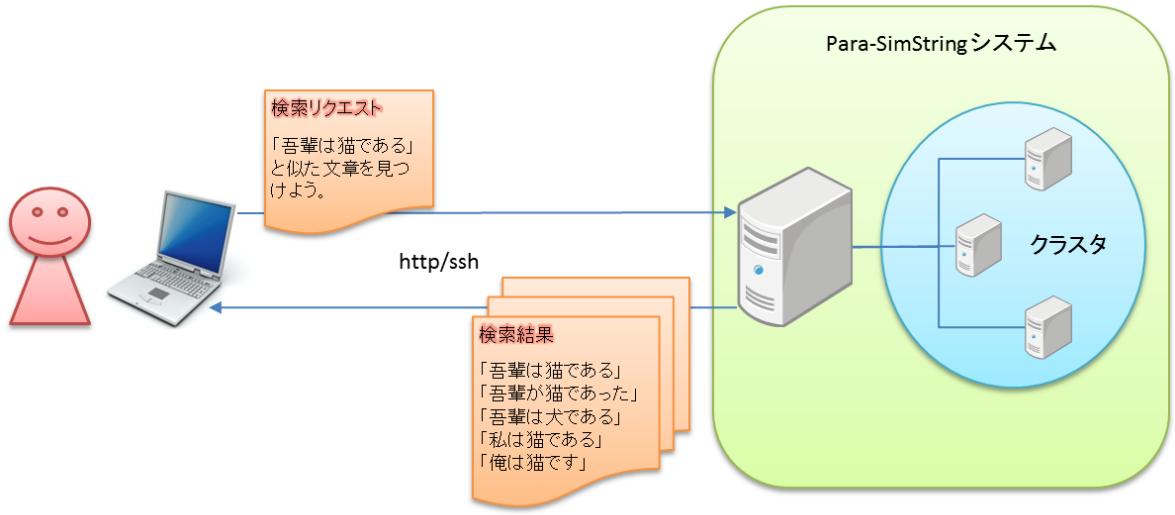

図 1 システムの使用イメージ

Para-SimString システムは検索システムです。検索といっても完全一致や部分一致検索ではなく、類似文字列検索を行うためのシステムです。

ユーザがシステムを使用するには、Web ブラウザで Para-SimString の検索ページにアクセスするか、SSH コマンドでサーバにログオンする必要があります。何れかの方法でシステムにアクセスしたユーザは、検索したい文字列を指定して検索リクエストを送信します。

システムには検索対象のテキストデータが、事前にインデックス化されて保存されています。その中からユーザが検索リクエストに指定した文字列と類似する文を探し出します。この時の比較は改行で区切られた行単位で行われ、類似していると判断されればその行が検索結果の一部となります。全ての結果が揃った時点で、システムは検索結果を返します。

1-1 パッケージ内容

Para-SimString システムは１つのパッケージファイル(Para-SimSting-SC-OSS-201203.tar.gz)にアーカイブされています。このパッケージファイルは tar コマンドにより解凍することができます。解凍後のパッケージの簡単な内容を以下に記します。

表 1 スクリプトモジュールの構成

パス	説明
Para-SimString/	Para-SimString システムを格納するディレクトリです。パッケージファイルを解凍するとこのディレクトリが復元されます。
data-sample/	検索対象となるテキストとして用意されたサンプルデータが格納されています。後述するシステム構築例で使用されています。
scripts/	スクリプトモジュールを格納しています。
simstring-1.0/	SimString 1.0 を格納しています。
web/	Web アプリケーションモジュールを格納しています。
Para-SimString-manual.pdf	Para-SimString システムマニュアル（本書）です。
query.txt	クエリーファイルのサンプルです。
server.list	デフォルトサーバリストファイルのサンプルです。

1-2 SimString について

Para-SimString システムで使用している SimString とは、岡崎直観氏が開発した類似文字列検索ライブラリです。SimString が実現している類似文字列検索は、クエリ文字列と類似するものを文字列集合の中から見つけ出す操作のことです。対象がクエリ文字列と完全に一致している必要はありません。検索する際には類似度を評価する関数を指定することができます。SimString ではコサイン係数、ジャッカード係数、ダイス係数、オーバーラップ係数に対応しています。

詳細は以下のウェブサイトを参照してください。

Naoaki Okazaki's website（岡崎直観氏のウェブサイト）
http://www.chokkan.org/

岡崎直観氏のウェブサイトの SimString のページ
http://www.chokkan.org/software/simstring/

2 構築計画の立案

Para-SimString のシステムアーキテクチャはクラスタ環境での動作を考慮しています。単体のサーバによる構成とは異なるため、事前に導入計画を立てておくことが重要です。まずはこの章の内容を把握し、システムがどういった要素で構成されるのかを理解してください。

2-1 システムの構成要素

Para-SimString システムはネットワークでつながった複数のサーバから構成されます。基本的には1台のハブサーバと複数台のノードサーバから構成されます。

2-1-1 ハブサーバ

ハブサーバはシステムの要となるサーバで、各ノードサーバをコントロールします。Web インターフェースを利用する場合はハブサーバ上で Web サーバが動かす必要があります。

2-1-2 ノードサーバ

ノードサーバは実際にインデックス作成処理や検索処理を行うサーバです。インデックス作成処理や検索処理は対象のデータサイズが大きくなると非常に時間が掛かります。その場合にそれらの処理を複数台のノードサーバに分散させて行うと時間を短縮させることができます。

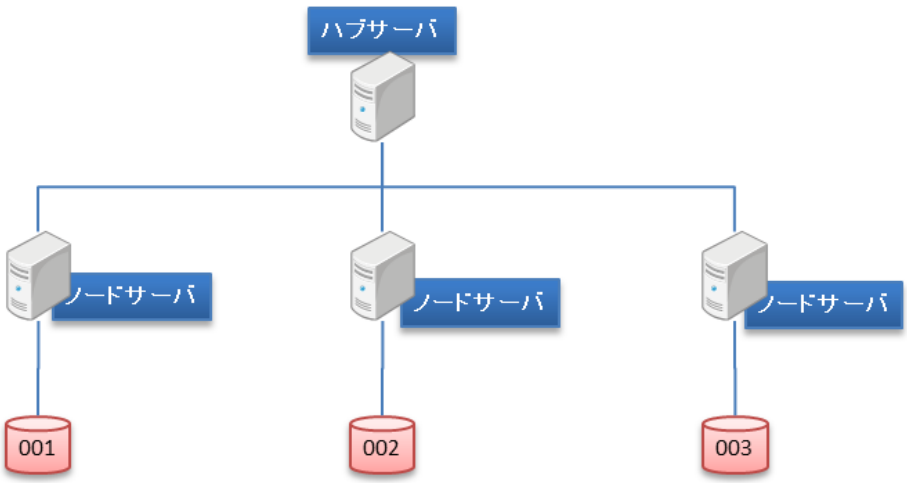

図 2 基本的なシステム構成要素

２-２　アーキテクチャ

このマニュアルが対象としている Para-SimString システムは、ハブサーバとノードサーバの台数によって以下の３種類のアーキテクチャをとることができます。

２-２-１　ノードサーバが複数台の構成

この構成は１台のハブサーバと２台以上のノードサーバでシステムを構成します。これは最も一般的な構成です。本書の説明もこの構成を前提として記載しています。

検索対象として大きなデータを扱う際にはそのサイズに応じてノードサーバを増やし、１台のノードサーバに掛かる負荷を減らすようにシステムを構築します。

各ノードサーバのマシンスペックや実現したい応答性能にもよりますが、大体１台のノードサーバが取り扱うデータサイズがテキストデータ（インデックスではない）の状態で数百 MB～数 GB 程度となるように構成することをお奨めします。

２-２-２　ノードサーバが１台の構成

この構成は１台のハブサーバと１台のノードサーバでシステムを構成します。これはインデックス作成処理や検索処理の負荷分散の観点からは意味のない構成です。ただし、取り扱うデータサイズが比較的小さい場合は考慮に値する構成です。例えば、検索対象のテキストが 500MB を超えない場合等です。それ以上大きいデータを扱う場合は、ノードサーバを複数台にする構成をお奨めします。

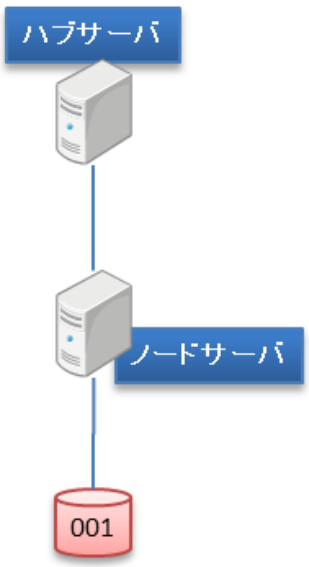

図 3　ノードサーバ１台の構成

2-2-3 スタンドアロンサーバが１台の構成

この構成はハブサーバとノードサーバを１台のサーバに集約してシステムを構成します。これは負荷分散との観点からは上記の**「2-2-2 ノードサーバが１台の構成」**と同様の欠点があります。それに加えて、インデックス作成処理や検索処理の負荷が Web サーバの処理にも影響を与える可能性があります。

取り扱うデータサイズが小さい場合でもこの構成はお奨めしませんが、システムの利用頻度が非常に少ない場合や実験的にシステムを試したい場合には採用されるかもしれません。その場合は、本書に記されているハブサーバとノードサーバに対して行う作業を、唯一のサーバに対して行う様に読み替えてください。

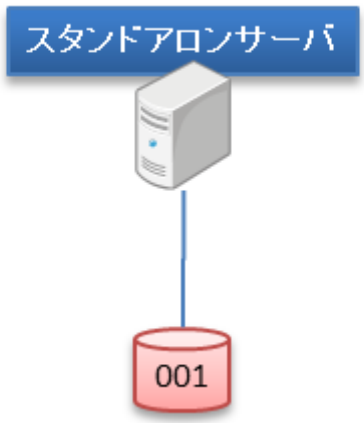

図 4 スタンドアロンサーバ構成

2-3 システムアーキテクチャの決定

システムの構成要素が理解できたら「実際のアーキテクチャをどうするのか」や「ノードサーバは何台にするのか」といったことを決めます。もちろんそれを決めるには、用意できるサーバのスペックも考えなければいけませんし、そもそも用意できるサーバの台数に限度がある場合もあるでしょう。それらを総合的に判断し、システムアーキテクチャの種類とノードサーバ台数を決めてください。

２-４ 検索対象データの分割

「２-２-１ ノードサーバが複数台の構成」を採用する場合は分散処理を行うということになります。この場合は処理される検索対象データを分割する必要があります。

最も簡単な分割方法は、検索対象テキストを均等に分割してノードサーバに割り振ることです。この場合の分割データ数はノードサーバ数に等しくなり、各々のデータサイズも同じくらいになります。

もう少し複雑な方法としては、ノードサーバの性能に応じて割り当てるデータサイズを変えるという方法や、ノードサーバ数より多くデータを分割しておき、１台のノードサーバに複数のデータを担当させる方法も考えられます。

ここでは最初に挙げた「最も簡単な分割方法」を推奨します。ノードサーバの性能は単純に CPU のクロック数だけで決まるわけではなく、大きなサイズのファイルを扱うための IO 性能やメモリサイズ、厳密にいうとネットワーク I/F の性能まで関わってきます。つまり、ノードサーバの性能の簡単な指標はないということです。この点において厳密さを求めるより、均等分割を行い、それであまりに性能差が出た時にデータの再構築を行うという方法をお奨めします。

インストールと設定

3 サーバの準備

システムに使用するサーバを準備します。以下に各種サーバ（ハブサーバ／ノードサーバ／スタンドアロンサーバ）が満たすべき条件を記します。

表 2 サーバ条件

項目	条件	備考	ハブサーバ	ノードサーバ	スタンドアロンサーバ
ハードウェア	Intel Xeon 5160 3.0GHz メインメモリ 4GB ネットワーク 1GbE	インデックス作成／検索を行うノードサーバの推奨スペック。さらにデータを格納できるだけの HDD 空き容量が必要。 ハブサーバは Web サーバが快適に動作する程度のスペックが必要。	✔	✔	✔
オペレーティングシステム	Cent OS 5（推奨）	以下の条件を満たすなら、他の Linux ディストリビューションでも動作可能。	✔	✔	✔
コマンドラインシェル	Bash	“/bin/bash”を想定。左記のパス以外の場合、Bash スクリプト内のパスを書き換えて対応可能。 また”LANG=ja_JP.UTF-8”となっていること。	✔	✔	✔
Perl	Perl 5.8 以上	“/usr/bin/perl”を想定。左記のパス以外の場合、Perl スクリプト内のパスを書き換えて対応可能。	✔		✔
PHP	PHP 5.1 以上	Web インターフェースを使用する場合は必須。	✔		✔
Web サーバ	Apache HTTP Server 2.2 (推奨)	サーバサイドで上記の PHP スクリプトが実行可能なら他の Web サーバでも良い。 Web インターフェースを使用する場合は必須。	✔		✔
ssh コマンド利用環境	ssh コマンドが利用できること。 詳細は右記参照。	実行ユーザがそれぞれのノードサーバに ssh コマンドを使ってパスワードなしでログインできること。	✔	✔	✔

これらの条件に適合するサーバを構築計画で決めた台数分用意します。それらを適切にネットワークに接続します。

これ以降の説明は、これらのインフラが整っていること前提に記されています。

3-1 実行ユーザの設定

実行ユーザにはハブサーバの実行ユーザとノードサーバの実行ユーザがあります。

スタンドアロンサーバは自身に SSH ログインするので、ハブサーバの実行ユーザとノードサーバの実行ユーザの両方の設定を行う必要があります。

3-1-1 ハブサーバの実行ユーザ

ハブサーバでのインデックス作成や検索時に処理を実行するユーザです。

コマンドラインでスクリプト（インデックス作成か検索かに関わらず）を実行する際の実行ユーザはスクリプトを実行したユーザとなります。

Web インターフェースを使用した検索の際の実行ユーザは Web サーバの実行ユーザとなります。

何れの場合にもパスワードなしでノードサーバに SSH ログインできる必要があります。

3-1-2 ノードサーバの実行ユーザ

ノードサーバでのインデックス作成や検索時に処理を実行するユーザです。

コマンドラインによる操作か Web インターフェースによる操作かに関わらず、SSH ログインされたユーザがノードサーバの実行ユーザとなります。

3-1-3 実行ユーザの設定

ハブサーバにおいてコマンドラインスクリプトの実行ユーザを決め、必要ならユーザの追加を行います。ユーザの追加方法はそのオペレーティングシステムに適切な方法で行ってください。

さらに Web サーバの実行ユーザを決め、必要ならユーザの追加と Web サーバの実行ユーザの設定を行います。これらもそのオペレーティングシステムや Web サーバに適切な方法で行ってください。

上記のそれぞれのユーザが全てのノードサーバに対して、パスワードなしに SSH ログインできるように以下の設定をハブサーバ上で行います。

1. **ハブサーバ実行ユーザでログインします。**
2. **パスワードなしの鍵を作成します。**

```
$ ssh-keygen -N "" -t rsa
```

3. ノードサーバ（***NODE_SERVER***）の実行ユーザ（***NODE_USER***）に対して、作成した公開鍵を登録します。

```
$ ssh-copy-id -i ~/.ssh/id_rsa.pub NODE_USER@NODE_SERVER
```

全てのハブサーバ実行ユーザが全てのノードサーバ実行ユーザに対して、上記の処理を行います。

３-２　Web サーバの設定

Web サーバは Apache HTTP Server 2.2 を推奨しますが、PHP スクリプトを実行できる Web サーバであれば特に問題はありません。必要であれば Web サーバに設定を行ってください。その方法は Web サーバのドキュメントを参照してください。

ここでは Apache HTTP Server 2.2（以降 Apache と記す）を使用している場合の設定方法を記します。

３-２-１　実行ユーザの設定

Apache の実行ユーザの設定は Apache の設定ファイル（“/etc/httpd/conf/httpd.conf”等）を開き、User/Group ディレクティブを確認してください。User にハブサーバ実行ユーザ名（***USER_NAME***）、Group にそのユーザのグループ名（***GROUP_NAME***）を指定します。

```
User USER_NAME
Group GROUP_NAME
```

3-2-2 システムを公開する URL 設定

システムの Web インターフェースを公開する URL を決め、それに対応するディレクトリを決めてください。公開 URL を“http://HOST/searchsys/”として対応するディレクトリを“/home/searchsys/”とする場合は、Apache の設定ファイル（“/etc/httpd/conf/httpd.conf”等）に以下の記述を追加します。

```
Alias /searchsys/ ”/home/searchsys/”

<Directory "/home/searchsys">
   AllowOverride All
   Options None
   Order allow,deny
   Allow from all
</Directory>
```

3-3 ダウンロード

パッケージファイル(Para-SimSting-SC-OSS-201203.tar.gz)をダウンロードして入手します。入手したパッケージファイルは各サーバに保存しておきます。

4 ハブサーバの設定

ここではハブサーバの設定について説明します。

4-1 インストールパスの決定

システムをインストールする前に、スクリプトモジュールをインストールするパスを決めます。また、「3-2-2 システムを公開する URL 設定」で説明した様に、Web インターフェースを公開する URL を決定し、Web アプリケーションモジュールをインストールするパスを把握しておきます。

これ以降、スクリプトモジュールをインストールするパスを ***PSS_HOME***、Web アプリケーションモジュールをインストールするパスを ***PSS_WEB_HOME*** と表記します。

4-2 パッケージファイルの展開と配備

入手したパッケージファイルをインストール対象のサーバに適切な方法でコピーします。そして、パッケージファイルを展開して ***PSS_HOME*** として配置します。

```
$ tar zxvf Para-SimSting-SC-OSS-201203.tar.gz
$ mv Para-SimString PSS_HOME
```

展開したパッケージには Web アプリケーションモジュールが含まれています。それをディレクトリごと ***PSS_WEB_HOME*** として配置します。

```
$ cd PSS_HOME
$ mv web PSS_WEB_HOME
```

4-3 SimString のインストール

ハブサーバでは SimString のインストールは不要です。

4-4 スクリプトモジュールの設定

スクリプトモジュールの初期設定を行います。

4-4-1 初期化スクリプトの変更

PSS_HOME/scripts/lrs-init.sh の内容を動作環境に合わせて変更します。ハブサーバの場合は以下の様に設定します。

```
export PSS_HOME=PSS_HOME
export LANG=ja_JP.UTF-8
```

4-4-2 スクリプト実行権限の付与

実行ユーザが ***PSS_HOME***/scripts にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod 755 PSS_HOME/scripts/*
```

4-4-3 結果ディレクトリの作成

結果ディレクトリを作成し、実行ユーザがそこを使用できるように設定します。結果ディレクトリには複数のユーザの権限によって書き込みが行われます。処理が中断した際にファイルの消去漏れが発生し、残ったファイルを手動で消さなければならないことがあります。そこで、念のためにディレクトリのスティッキービットを立てておきます。こうすることにより、結果ディレクトリの所有者が残ったファイルの処理をすることができる様になります。

```
$ mkdir PSS_HOME/results
$ chmod 1777 PSS_HOME/results
```

4-4-4 データディレクトリの作成

ハブサーバではデータディレクトリの作成は不要です。

４-４-５ サーバリストの設定

構築計画での決定に基づいてデフォルトサーバリストを編集します。デフォルトサーバリストは ***PSS_HOME***/server.list です。

サーバリストの書式を以下に記します。

```
[USER@]NODE_SERVER:NODE_PSS_HOME:DATA_ID↓
                    .
                    .
                    .
```

１行毎にあるノードサーバに対する設定となっています。上記の意味は ***NODE_SERVER*** というホスト名（または IP アドレス）のノードサーバに ***USER*** という名前のユーザが SSH でログインする際の設定です。ユーザ名がローカルとリモートで同じなら ***USER***@の部分は省略できます。

NODE_PSS_HOME は ***NODE_SERVER*** における ***PSS_HOME*** です。

DATA_ID は処理（インデックス作成、または、検索）の対象となるデータ ID です。

これらを分割した検索対象データ全てに対して指定します。もし指定されなかった場合は、そのデータ ID のデータは処理の対象となりません。

サーバリストの具体的な例を以下に記します。この例では検索対象データは４分割されました。しかし、ハブサーバ用のサーバ以外にノードサーバ用のサーバは３台（node001.mylab.jp, node002.mylab.jp, node003.mylab.jp）しか都合がつきませんでした。残り一台（test.otherlab.jp）は他の研究室のテストサーバを間借りすることにしました。

ハブサーバと都合のついた３台のサーバは root 権限で環境構築ができたため“/usr/local/pss”をアプリケーションのホームディレクトリ、実行ユーザは“pss”ユーザに統一して環境を構築しました。これによりサーバリストのユーザ指定は省略できます。処理するデータは node001.mylab.jp にデータ ID が 001 のデータを、node002.mylab.jp にデータ ID が 002 のデータを、node003.mylab.jp にデータ ID が 003 のデータをそれぞれ割り当てました。

残りの一台は間借りサーバのため、“testuser”アカウントを作ってもらい、そのユーザのホームディレクトリ“/home/testuser”に環境を構築することになりました。処理するデータのデータ ID が 004 のデータを割り当てました。

```
node001.mylab.jp:/usr/local/pss:001
node002.mylab.jp:/usr/local/pss:002
node003.mylab.jp:/usr/local/pss:003
testuser@test.otherlab.jp:/home/testuser:004
```

Web アプリケーションモジュールの設定

Web アプリケーションモジュールの初期設定を行います。

4-4-6 初期化スクリプトの変更

PSS_WEB_HOME/lrs-init.php の内容を動作環境に合わせて変更します。変更する箇所は$PSSHome の初期値のみです。

```
$PSSHome = 'PSS_HOME'
```

4-4-7 スクリプト実行権限の付与

Web サーバ実行ユーザが ***PSS_WEB_HOME*** にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod -R 755 PSS_WEB_HOME
```

4-4-8 Web サーバの起動

最後に Web サーバを起動してこれらの設定を反映させます。もし起動済みなら再起動を行います。以下は Cent OS 5 での Web サーバ起動の例です。

```
$ sudo /etc/init.d/httpd start
```

5 ノードサーバの設定

ここではノードサーバの設定について説明します。

5-1 インストールパスの決定

システムをインストールする前に、スクリプトモジュールをインストールするパスを決めます。これ以降、スクリプトモジュールをインストールするパスを ***PSS_HOME*** と表記します。

ノードサーバでは Web アプリケーションモジュールはインストールしません。

5-2 パッケージファイルの展開と配備

入手したパッケージファイルをインストール対象のサーバに適切な方法でコピーします。そして、パッケージファイルを展開して ***PSS_HOME*** として配置します。

```
$ tar zxvf Para-SimSting-SC-OSS-201203.tar.gz
$ mv Para-SimString PSS_HOME
```

展開したパッケージには Web アプリケーションモジュールが含まれていますが、ノードサーバでは使用しませんので削除しても構いません。

5-3 SimString のインストール

類似文字列検索ライブラリ SimString をインストールします。上記で Para-SimString のパッケージファイルを展開していれば、***PSS_HOME***/simstring-1.0 ディレクトリに SimString のバージョン 1.0 があります。また、以下のウェブサイトから SimString の最新版が入手できます。

Naoaki Okazaki's website の SimString ページ URL：

http://www.chokkan.org/software/simstring/index.html.ja

また、インストール方法もこのサイトを参照してください。

これ以降 SimString コマンドのインストールパスを ***SIM_STRING_PATH***/simstring と記します。この値は Para-SimString の設定の際に必要になりますので覚えておいてください。

5-4 スクリプトモジュールの設定

スクリプトモジュールの初期設定を行います。

5-4-1 初期化スクリプトの変更

PSS_HOME/scripts/lrs-init.sh の内容を動作環境に合わせて変更します。ノードサーバの場合は以下の様に設定します。

```
export PSS_HOME=PSS_HOME
export PATH_SIMSTRING=SIM_STRING_PATH/simstring
export LANG=ja_JP.UTF-8
```

5-4-2 スクリプト実行権限の付与

実行ユーザが ***PSS_HOME***/scripts にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod 755 PSS_HOME/scripts/*
```

5-4-3 結果ディレクトリの作成

結果ディレクトリを作成し、実行ユーザがそこを使用できるように設定します。結果ディレクトリには複数のユーザの権限によって書き込みが行われます。処理が中断した際にファイルの消去漏れが発生し、残ったファイルを手動で消さなければならないことがあります。そこで、念のためにディレクトリのスティッキービットを立てておきます。こうすることにより、結果ディレクトリの所有者が残ったファイルの処理をすることができる様になります。

```
$ mkdir PSS_HOME/results
$ chmod 1777 PSS_HOME/results
```

5-4-4 データディレクトリの作成

データディレクトリを作成し、実行ユーザがそこを使用できるように設定します。***DATA_ID*** はこのサーバに割り当てられた検索対象のデータ ID を表します。結果ディレクトリと同様の理由でデータ ID のディレクトリにもスティッキービットを立てておきます。

```
$ mkdir -p PSS_HOME/data/DATA_ID/src
$ chmod 1777 PSS_HOME/data/DATA_ID/
$ chmod 755 PSS_HOME/data/DATA_ID/src
```

上記の設定ができたら ***PSS_HOME***/data/***DATA_ID***/src に対応するデータ ID の検索対象テキストを格納します。その際、検索対象テキストは検索対象としたい行単位に改行（LF）で分割しておきます。

5-4-5 サーバリストの設定

ノードサーバではサーバリストは使用しませんので設定は不要です。

5-5 Web アプリケーションモジュールの設定

ノードサーバでは Web アプリケーションモジュールの設定は不要です。

6 スタンドアロンサーバの設定

ここではスタンドアロンサーバの設定について説明します。

6-1 インストールパスの決定

システムをインストールする前に、スクリプトモジュールをインストールするパスを決めます。また、「3-2-2 システムを公開する URL 設定」で説明した様に、Web インターフェースを公開する URL を決定し、Web アプリケーションモジュールをインストールするパスを把握しておきます。

これ以降、スクリプトモジュールをインストールするパスを ***PSS_HOME***、Web アプリケーションモジュールをインストールするパスを ***PSS_WEB_HOME*** と表記します。

6-2 パッケージファイルの展開と配備

入手したパッケージファイルをインストール対象のサーバに適切な方法でコピーします。そして、パッケージファイルを展開して ***PSS_HOME*** として配置します。

```
$ tar zxvf Para-SimSting-SC-OSS-201203.tar.gz
$ mv Para-SimString PSS_HOME
```

展開したパッケージには Web アプリケーションモジュールが含まれています。それをディレクトリごと ***PSS_WEB_HOME*** として配置します。

```
$ cd PSS_HOME
$ mv web PSS_WEB_HOME
```

6-3 SimString のインストール

類似文字列検索ライブラリ SimString をインストールします。上記で Para-SimString のパッケージファイルを展開していれば、***PSS_HOME***/simstring-1.0 ディレクトリに SimString のバージョン 1.0 があります。また、以下のウェブサイトから SimString の最新版が入手できます。

Naoaki Okazaki's website の SimString ページ URL：

http://www.chokkan.org/software/simstring/index.html.ja

また、インストール方法もこのサイトを参照してください。

これ以降 SimString コマンドのインストールパスを ***SIM_STRING_PATH***/simstring と記します。この値は Para-SimString の設定の際に必要になりますので覚えておいてください。

6-4 スクリプトモジュールの設定

スクリプトモジュールの初期設定を行います。

6-4-1 初期化スクリプトの変更

PSS_HOME/scripts/lrs-init.sh の内容を動作環境に合わせて変更します。スタンドアロンサーバの場合は以下の様に設定します。

```
export PSS_HOME=PSS_HOME
export PATH_SIMSTRING=SIM_STRING_PATH/simstring
export LANG=ja_JP.UTF-8
```

6-4-2 スクリプト実行権限の付与

実行ユーザが ***PSS_HOME***/scripts にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod 755 PSS_HOME/scripts/*
```

6-4-3 結果ディレクトリの作成

結果ディレクトリを作成し、実行ユーザがそこを使用できるように設定します。結果ディレクトリには複数のユーザの権限によって書き込みが行われます。処理が中断した際にファイルの消去漏れが発生し、残ったファイルを手動で消さなければならないことがあります。そこで、念のためにディレクトリのスティッキービットを立てておきます。こうすることにより、結果ディレクトリの所有者が残ったファイルの処理をすることができる様になります。

```
$ mkdir PSS_HOME/results
$ chmod 1777 PSS_HOME/results
```

6-4-4 データディレクトリの作成

データディレクトリを作成し、実行ユーザがそこを使用できるように設定します。***DATA_ID*** はこのサーバに割り当てられた検索対象のデータ ID を表します。結果ディレクトリと同様の理由でデータ ID のディレクトリにもスティッキービットを立てておきます。

スタンドアロンサーバの場合、データの分割が必要なければ ***DATA_ID*** は”001”だけで良いでしょう。

```
$ mkdir -p PSS_HOME/data/DATA_ID/src
$ chmod 1777 PSS_HOME/data/DATA_ID/
$ chmod 755 PSS_HOME/data/DATA_ID/src
```

上記の設定ができたら ***PSS_HOME***/data/***DATA_ID***/src に対応するデータ ID の検索対象テキストを格納します。その際、検索対象テキストは検索対象としたい行単位に改行（LF）で分割しておきます。

6-4-5 サーバリストの設定

構築計画での決定に基づいてデフォルトサーバリストを編集します。デフォルトサーバリストは ***PSS_HOME***/server.list です。

サーバリストの書式を以下に記します。

```
[USER@]NODE_SERVER:NODE_PSS_HOME:DATA_ID↓
                   .
                   .
                   .
```

１行毎にあるノードサーバに対する設定となっています。上記の意味は ***NODE_SERVER*** というホスト名（または IP アドレス）のノードサーバに ***USER*** という名前のユーザが SSH でログインする際の設定です。ユーザ名がローカルとリモートで同じなら ***USER***@の部分は省略できます。スタンドアロンサーバはローカルとリモートが同じなので、特別な理由がない限り ***USER***@の部分は必要がなく、***NODE_SERVER*** は“localhost”となるでしょう。

NODE_PSS_HOME は ***NODE_SERVER*** における ***PSS_HOME*** です。

DATA_ID は処理（インデックス作成、または、検索）の対象となるデータ ID です。

これらを分割した検索対象データ全てに対して指定します。もし指定されなかった場合は、そのデータ ID のデータは処理の対象となりません。スタンドアロンサーバはデータを分割する意味がないため、***DATA_ID*** は 001 だけとなることが多いでしょう。

スタンドアロンサーバで検索対象を分割せず、かつ、SSH ログインユーザを自分自身にする場合には、以下の 1 行のみの設定となります。

```
localhost:PSS_HOME:001
```

6-5 Web アプリケーションモジュールの設定

Web アプリケーションモジュールの初期設定を行います。

6-5-1 初期化スクリプトの変更

PSS_WEB_HOME/lrs-init.php の内容を動作環境に合わせて変更します。変更する箇所は$PSSHome の初期値のみです。

```
$PSSHome = 'PSS_HOME'
```

6-5-2 スクリプト実行権限の付与

Web サーバ実行ユーザが ***PSS_WEB_HOME*** にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod -R 755 PSS_WEB_HOME
```

6-5-3 Web サーバの起動

最後に Web サーバを起動してこれらの設定を反映させます。もし起動済みなら再起動を行います。以下は Cent OS 5 での Web サーバ起動の例です。

```
$ sudo /etc/init.d/httpd start
```

7 インデックスの作成

検索はテキストそのものではなく、それらをインデックス化したデータベースに対して行われます。ここでは検索対象のテキストをインデックス化する方法を記します。

既に**「5-4-4 データディレクトリの作成」**で記されている様に、検索対象のテキストはデータディレクトリに格納されています。その際、検索対象テキストは検索対象としたい行単位に改行（LF）で分割しておかなければなりません。SimString は検索対象を改行区切りの行単位で検索するため、適切な位置で改行を行っていなければ、適切な検索結果とならないことに注意してください。

加えて「4-4-5 サーバリストの設定」で記されている様に、どのノードサーバがどのデータを取り扱うかという情報がデフォルトサーバリストに記載されているかを確認してください。サーバリストの準備ができていない場合はインデックス作成前に用意してください

インデックス作成の準備が整ったら、システム管理者がハブサーバ（スタンドアロンサーバ）にログインして、次の様にインデックス作成スクリプトを実行します。

```
$ PSS_HOME/scripts/lrs-index.sh
```

7-1 サーバリストファイルの指定

もしインデックス作成にデフォルトサーバリストを使わず、他のサーバリストを使用したい場合は次の様にインデックス作成スクリプトを実行します。

```
$ PSS_HOME/scripts/lrs-index.sh -L SERVER_LIST_FILE
```

SERVER_LIST_FILE にサーバリストのファイルパスを指定します。ここにデフォルトサーバリスト以外のファイルを指定した場合は、そのファイルをデフォルトサーバリストと置き換えて新しいデフォルトサーバリストにすべきです。そうしなかった場合、検索時にノードサーバとインデックスの対応に食い違いが発生するかもしれず、その結果、適切に検索処理を実行できなくなる可能性があります。

7-2 実行と結果

インデックス作成スクリプトは完了するまでに数分～数時間掛かる場合があります。サーバのスペックや検索対象テキストの内容にも依りますが、推奨スペックのサーバでは 500MB のテキストをインデックス化するのに 20 分程度は必要となります。

エラーメッセージが表示されずに”Completed”メッセージが表示されればインデックスの作成は成功です。これで検索を行う準備が整いました。

8 スタンドアロンサーバ構築例

ここではスタンドアロンサーバの構築例を記します。

Para-SimString パッケージには検索対象のサンプルデータ（このマニュアルの内容をテキストにしたもの）が付随しています。この例ではそのサンプルデータを検索対象とする検索システムを構築します。

スタンドアロンサーバなので使用するサーバは 1 台です。この例ではそのサーバのホスト名を”pss-server”として説明をします。

また、特別に記載されていない限り、全ての作業は Para-SimString の管理ユーザとして用意した”pssadmin”というユーザで行われているものとします。

8-1 Para-SimString パッケージのインストール

今回、Para-SimString を”/home/pssadmin/para-simstring”にインストールすることに決めました。このディレクトリのパスが ***PSS_HOME*** になります。

次に、Web 公開の URL は”http://pss-server/pssadmin/para-simstring/”に決めました。この URL に該当するディレクトリは”/home/pssadmin/public_html/para-simstring”でした。このディレクトリのパスが ***PSS_WEB_HOME*** になります。

pssadmin ユーザとして pss-server にログインした後、入手した Para-SimString パッケージファイルを展開して ***PSS_HOME*** に配置します。

```
$ tar zxvf Para-SimSting-SC-OSS-201203.tar.gz
$ mv Para-SimString /home/pssadmin/para-simstring
```

次に、展開したパッケージ内の Web アプリケーションモジュールディレクトリを ***PSS_WEB_HOME*** に配置します。

```
$ cd /home/pssadmin/para-simstring
$ mv web /home/pssadmin/public_html/para-simstring
```

SimString をインストールします。インストールの詳細は割愛しますが、この例では以下の手順でインストールしたものとします。

```
$ cd simstring-1.0/
$ ./configure
$ make
$ make install
```

上記の結果、/home/pssadmin/para-simstring/simstring-1.0/frontend/simstring に SimString コマンドが作成されました。この例ではこのコマンドをそのまま使用することにします。

ここからは各種設定を行います。

scripts/lrs-init.sh を変更します。赤字が変更箇所です。

```
export PSS_HOME=/home/pssadmin/para-simstring
export PATH_SIMSTRING=/home/pssadmin/para-simstring/simstring-1.0/frontend
/simstring
export LANG=ja_JP.UTF-8
```

想定されるユーザがスクリプトを実行できるように、ファイルのパーミッション設定を行います。

```
$ chmod 755 scripts/*
```

結果ディレクトリを作成し、想定されるユーザがそこを使用できるように設定します。

```
$ mkdir results
$ chmod 1777 results
```

8-2 データディレクトリの作成

データディレクトリを作成します。今回はスタンドアロンサーバなので pss-server に唯一のデータディレクトリが検索対象となります。当然、データ ID は”001”だけです。

今回の検索対象であるサンプルデータは、十数ファイルに分割して ***PSS_HOME***/data-sample ディレクトリに保存してあります。ノードサーバが複数台ある（つまりデータ ID も複数ある）場合は、これらのファイルをそれぞれのデータ ID ディレクトリに分配することになりますが、今回はスタンド

アロンサーバ自身のデータ ID ディレクトリが 1 つしかないため、全てのファイルをそこに格納します。

```
$ mkdir -p data/001/src
$ cp -r data-sample/* data/001/src/
$ chmod 1777 data/001/
$ chmod 755 data/001/src
$ chmod 644 data/001/src/*
```

８-３ サーバリストの設定

デフォルトサーバリストを編集します。スタンドアロンサーバなので、デフォルトサーバリストの設定は以下の 1 行のみとなるでしょう。

```
localhost:/home/pssadmin/para-simstring:001
```

８-４ Web アプリケーションモジュールの設定

Web アプリケーションモジュールの初期設定を行うので作業ディレクトリを移動します。

```
$ cd /home/pssadmin/public_html/para-simstring
```

lrs-init.php の$PSSHome の初期値を変更します。

```
$PSSHome = '/home/pssadmin/para-simstring'
```

Web サーバ実行ユーザが ***PSS_WEB_HOME*** にある全てのスクリプトファイルを実行できる様に、パーミッションの設定を行います。

```
$ chmod -R 755 .
```

最後に Web サーバを起動してこれらの設定を反映させます。もし起動済みなら再起動を行います。以下は Cent OS 5 での Web サーバ起動の例です。

```
$ sudo /etc/init.d/httpd start
```

８-５ インデックスの作成

ここまででシステムの設定は終わりました。実際に検索を行うためには、インデックスを作成しなければいけません。

```
$ cd /home/pssadmin/para-simstring
$ ./scripts/lrs-index.sh
```

インデックスの作成が終わるまで、恐らく 1 分もかからないでしょう。エラーメッセージが表示されずに”Completed”メッセージが表示されればインデックスの作成は成功です。以降の使用方法を読んで、検索してみてください。

使用方法

9 検索の実行

ここでは実際の検索を行う方法について説明します。

検索にはコマンドラインで検索スクリプトを実行する方法と、Web インターフェースを使って実行する方法があります。

9-1 コマンドライン検索

インデックスの作成に成功したら検索ができるようになります。検索はコマンドラインと Web インターフェースのどちらからでも行えます。ここではコマンドラインでの検索処理について記します。

インデックスの作成完了後、検索を実行するユーザはハブサーバ（スタンドアロンサーバ）にログインして、次の様に検索スクリプトを実行します。

```
$ PSS_HOME/scripts/lrs-search.sh QUERY_FILE SIM_STRING_OPTS
```

9-1-1 クエリファイルの指定

QUERY_FILE に検索する文字列を記したクエリファイルのパスを指定します。クエリファイル中には検索する文字列を複数指定でき、その場合は改行区切りで（つまり複数行にわたって）指定します。クエリファイルの文字エンコーディングは UTF-8 で、改行コードは LF です。

以下にクエリファイルの例を記します。

```
春はあけぼの↓
吾輩は猫である↓
      ・
      ・
      ・
鳴かぬなら鳴くまで待とう
```

9-1-2 SimString オプションの指定

SIM_STRING_OPTS に simstring コマンドのオプションを指定します。指定できるオプションは主に以下の 2 つです。

`-s similarity`

類似度評価関数の指定

similarity には次の何れかを指定します。デフォルトは cosine です。

指定	意味
exact	exact match
dice	dice coefficient
cosine	cosine coefficient
jaccard	jaccard coefficient
overlap	overlap coefficient

`-t threshold`

閾値の指定

類似度評価の閾値として threshold に 0.0~1.0 の値を指定します。閾値が高い程、より類似している文字列だけがヒットします。デフォルトは 0.7 です。

9-1-3 サーバリストファイルの指定

もし検索にデフォルトサーバリストを使わず他のサーバリストを使用したい場合は、インデックス作成時と同様に次の様に検索を実行できますが、サーバリストを変えるとノードサーバとインデックスの対応に食い違いが発生する可能性があるため推奨いたしません。

```
$ PSS_HOME/scripts/lrs-index.sh -L SERVER_LIST_FILE QUERY_FILE
SIM_STRING_OPTS
```

9-1-4 実行と結果

インデックスの大きさとクエリ文字列にもよりますが、検索は数秒～数分で終了します。検索が終了したら検索結果が標準出力に出力されます。出力結果は以下の様なフォーマットになります。

```
春はあけぼの     <=== 第 1 の検索キーワード
        春はあけぼの     <=== ヒットしたセンテンス
        春はあけぼの
        ―春はあけぼの
        ―春はあけぼの
                4 strings retreived    <=== ヒットした件数
我輩は猫である     <=== 第 2 の検索キーワード
        我輩は猫である     <=== ヒットしたセンテンス
                1 strings retreived    <=== ヒットした件数
```

9-2 Web インターフェースによる検索

インデックスの作成に成功したら検索ができるようになります。検索はコマンドラインと Web インターフェースのどちらからでも行えます。ここでは Web インターフェースでの検索処理について記します。

インデックスの作成完了後、検索を実行するユーザは任意の PC から Web ブラウザを使って「3-2-2 システムを公開する URL 設定」で決めた検索画面の URL にアクセスします。

9-2-1 クライアントの条件

オペレーティングシステムの条件は特にありませんが、現時点（2012 年 2 月）で比較的新しいバージョンの Web ブラウザを利用して Web サーバにアクセスする必要があります。

本システムは以下に記す 5 種類の Web ブラウザで動作確認を行っているため、それらを利用することを推奨します。

推奨 Web ブラウザ

- Windows Internet Explorer 9
- Mozilla Firefox 10
- Google Chrome 17
- Safari 5
- Opera 11

9-2-2 検索画面の操作

Web インターフェースの URL を Web ブラウザに指定して検索画面にアクセスします。検索画面は図の様になっています。各入力欄に適切な値を入力して検索を行います。

① クエリー文字列入力欄

類似文字列検索の検索文字列を入力します。

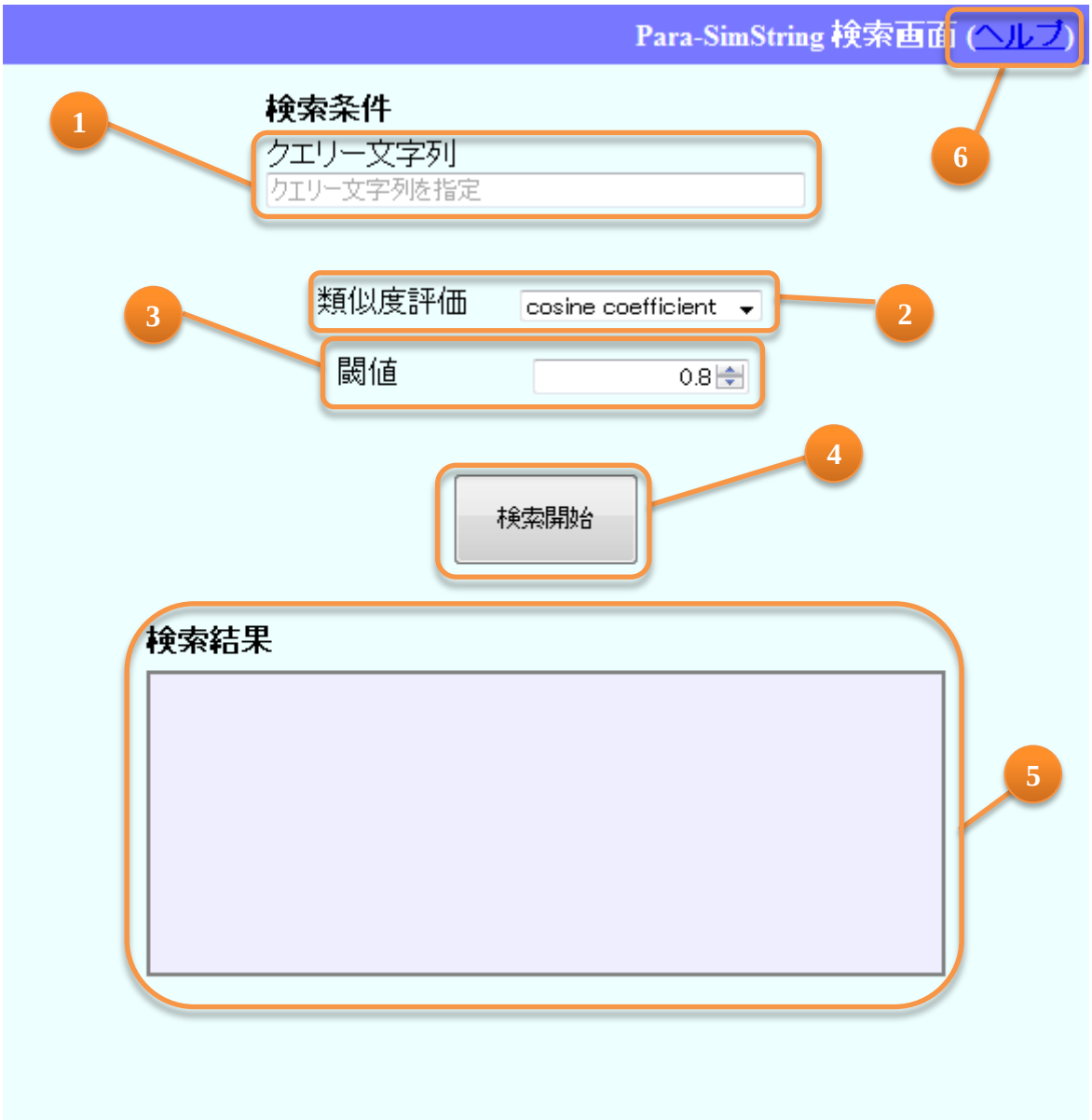

図 5 検索画面

② 類似度評価方式入力欄

類似文字列検索を行う際の類似度評価方式を選択します。選択項目として以下の項目があります。初期値は“cosine coefficient”です。

- exact match
- dice coefficient
- cosine coefficient (default)
- jaccard coefficient
- overlap coefficient

③ 閾値入力欄

類似文字列検索を行う際の類似度閾値を入力します。0.0~1.0 の範囲の実数が指定できます。値を大きく指定する程クエリー文字列と相違がない文字列がヒットするようになります。値を小さく指定する程クエリー文字列と相違がある文字列もヒットするようになります。初期値は“0.8”です。

④ 検索開始／検索中断ボタン

初期状態ではこのボタンには“検索開始”と表示されています。その時にこのボタンを押下すると、指定された条件で検索が始まります。検索が始まるとこのボタンのラベルは“中断”に変化します。

“中断”と表示されている時にこのボタンを押下すると、実行中の検索が中断します。検索が完了したり中断したりすると、このボタンのラベルは“検索開始”に変化します。

⑤ 検索結果表示欄

検索開始ボタンを押してから何らかの検索結果が出るまでの検索処理実行中、検索結果表示欄の場所には実行中を表すアニメーション表示と進捗率が表示されます。

検索が完了することで処理が停止すると、再び検索結果表示欄が表示されます。検索に成功した場合は検索結果表示欄に検索結果が表示されます。検索結果の表示形式は以下のとおりです。

```
春はあけぼの      <===検索キーワード
    春はあけぼの      <===ヒットしたセンテンス
    春はあけぼの
    —春はあけぼの
    —春はあけぼの
       4stringsretreived    <===ヒットした件数
```

図 6　検索処理実行中の表示

⑥ ヘルプリンク

このリンクはヘルプページへのリンクになっています。押下することでヘルプページが表示されます。

１０ ライセンス

Para-SimStringはフリーソフトウェアですが、著作権は独立行政法人情報通信研究機構に帰属します。
Para-SimString は、BSD ライセンス (Modified BSD License)、LGPL (GNU Lesser General Public License)、または、GPL (GNU General Public License) に従って使用、改変、再配布することができます。

また、Para-SimString システムは以下のソフトウェアを利用して作成されています。

１０-１ SimString

Para-SimString システムは岡崎直観氏の SimString 1.0 の利用を前提としており、頒布するパッケージ内に SimString 1.0 を含んでいます。
SimString 1.0 は修正 BSD ライセンスの下で利用しています。

```
The BSD license.

Copyright (c) 2009,2010 Naoaki Okazaki
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
    * Redistributions of source code must retain the above copyright
      notice, this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright
      notice, this list of conditions and the following disclaimer in the
      documentation and/or other materials provided with the distribution.
    * Neither the names of the authors nor the names of its contributors
      may be used to endorse or promote products derived from this
      software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

１０-２ jQuery

Para-SimString システムには jquery.com の jQuery 1.7.1 が含まれています。
jQuery 1.7.1 は MIT License の下で利用しています。

```
Copyright (c) 2011 John Resig, http://jquery.com/

Permission is hereby granted, free of charge, to any person obtaining
a copy of this software and associated documentation files (the
"Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish,
distribute, sublicense, and/or sell copies of the Software, and to
permit persons to whom the Software is furnished to do so, subject to
the following conditions:

The above copyright notice and this permission notice shall be
included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE
LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

以上