Molecular Profiling Research Center for Drug Discovery (MolProf), AIST

Semantic Analysis Service

User Manual

AIST

2015/07/24

1. Synchronous type SADI services 1
1.0. Uses of synchronous type SADI services 1
1.1. Blast 5
1.1.1. Preparing input RDF 5
1.1.2. Execution command 6
1.1.3. Execution result 7
1.2. CentroidFold 9
1.2.1. Preparing input RDF 9
1.2.2. Execution command 10
1.2.3. Execution result 11
1.3. ClustalW 13
1.3.1. Preparing input RDF 13
1.3.2. Execution command 14
1.3.3. Execution result 15
1.4. IPknot 17
1.4.1. Preparing input RDF 17
1.4.2. Execution command 18
1.4.3. Execution result 19
1.5. Mafft 20
1.5.1. Preparing input RDF 20
1.5.2. Execution command 22
1.5.3. Execution result 22
1.6. Psipred 24
1.6.1. Preparing input RDF 24
1.6.2. Execution command 25
1.6.3. Execution result 26
1.7. Raccess 28
1.7.1. Preparing input RDF 28
1.7.2. Execution command 29
1.7.3. Execution result 30
1.8. RactIP 32
1.8.1. Preparing input RDF 32
1.8.2. Execution command 33
1.8.3. Execution result 34
1.9. Wolfpsort 36
1.9.1. Preparing input RDF 36

1.9.2. Execution command 37
1.9.3. Execution result 38
2. Asynchronous type SADI services 39
2.0. Uses asynchronous type SADI services 39
2.1. Last 41
2.1.1. Preparing input RDF 41
2.1.2. Execution command 43
2.1.3. Execution result 45
2.2. Modelling 47
2.2.1. Preparing input RDF 47
2.2.2. Execution command 51
2.2.3. Execution result 52
2.3. PoodleL 54
2.3.1. Preparing input RDF 54
2.3.2. Execution command 55
2.3.3. Execution result 56
2.4. PoodleS 57
2.4.1. Preparing input RDF 57
2.4.2. Execution command 58
2.4.3. Execution result 59
2.5. Rassie 60
2.5.1. Preparing input RDF 60
2.5.2. Execution command 62
2.5.3. Execution result 63
Contact 65

Service		OWL classes corresponding to each input RDF
Blast	S	http://www.molprof.jp/ontologies/aistlssio.owl#BlastInput
CentroidFold	S	http://www.molprof.jp/ontologies/aistlssio.owl#CentroidFoldInput
ClustalW	S	http://www.molprof.jp/ontologies/aistlssio.owl#ClustalWInput
IPknot	S	http://www.molprof.jp/ontologies/aistlssio.owl#IPknotInput
Mafft	S	http://www.molprof.jp/ontologies/aistlssio.owl#MafftInput
Psipred	S	http://www.molprof.jp/ontologies/aistlssio.owl#PsiPredInput
Raccess	S	http://www.molprof.jp/ontologies/aistlssio.owl#RaccessInput
RactIP	S	http://www.molprof.jp/ontologies/aistlssio.owl#RactIPInput
Wolfpsort	S	http://www.molprof.jp/ontologies/aistlssio.owl#WolfPsortInput
Last	A	http://www.molprof.jp/ontologies/aistlssio.owl#LastInput
Modelling	A	http://www.molprof.jp/ontologies/aistlssio.owl#ModellingInput
PoodleL	A	http://www.molprof.jp/ontologies/aistlssio.owl#PoodleLInput
PoodleS	A	http://www.molprof.jp/ontologies/aistlssio.owl#PoodleSInput
Rassie	A	http://www.molprof.jp/ontologies/aistlssio.owl#RassieInput

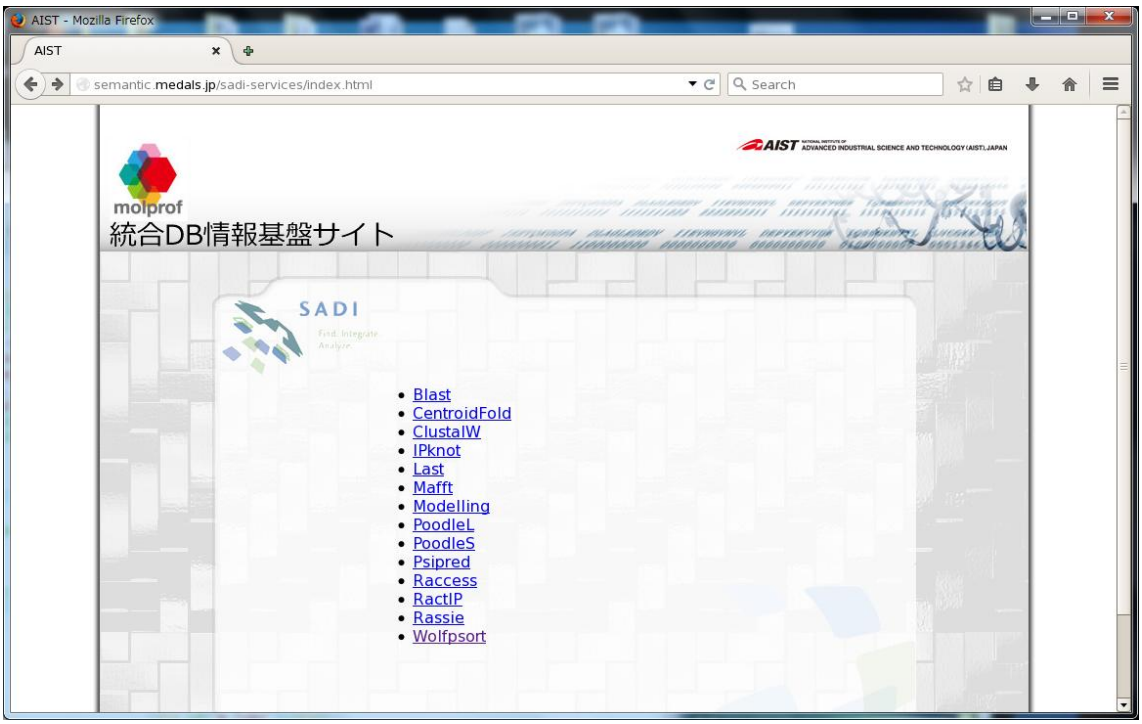
Figure 1-A Semantic web services and their input OWL class URLs

*** S: Synchronous, A: Asynchronous**

1. Synchronous type SADI services

1.0. Uses of synchronous type SADI services

The user can access “http://semantic.medals.jp:8090/sadi-services/index.html” and available SADI services are displayed on your web browser.



SADI services

Synchronous SADI services are Blast, CentroidFold, ClustalW, IPknot, Mafft, Psipred, Raccess, RactIP and Wolfpsort, and these services are executed by using the following cURL commands.

% curl --data-binary @”iput RDF file” http://semantic.medals.jp:8090/sadi-services/”SADI service name” (Figure 1-A) -o “output RDF file name”

If the user would like to execute Wolfpsort SADI service with an input.rdf and to get an output.rdf stored Wolfpsort service results,

% curl --data-binary @input.rdf
http://semantic.medals.jp:8090/sadi-services/Wolfpsort -o output.rdf

This input RDF format is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:WolfPsortInput
rdf:about="http://www.molprof.jp/ontologies/wolfpsort.rdf#1">
       <aistls:requiresKingdomInformationOf>animal</aistls:requiresKingdomInformationO
f>
       <aistls:requiresQueryProteinSequence>
    MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWG
QPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMD
EYSN
       </aistls:requiresQueryProteinSequence>
    </aistls:WolfPsortInput>
</rdf:RDF>
```

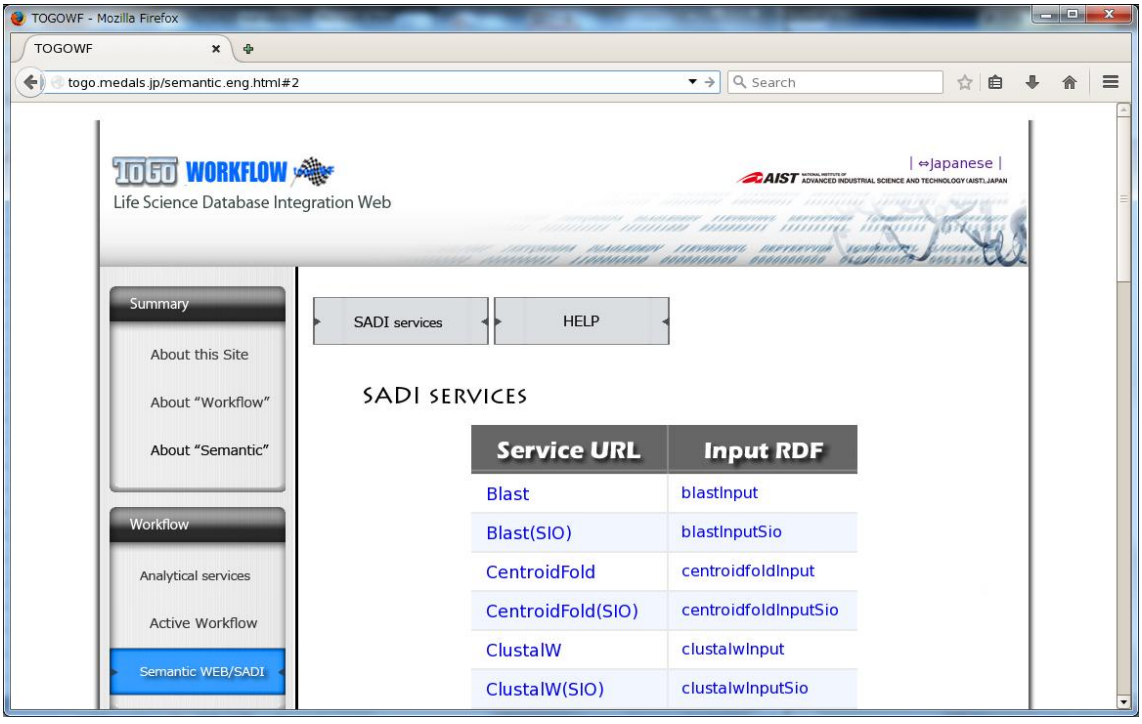
Input RDF for Wolfpsort SADI service

- Black: start and end tag of RDF
- Green: name spaces and their corresponding URLs
- Red: subject (input OWL class to execute a SADI service (Figure 1-A), and an arbitrary string
  e.g. WolfPsort
  input OWL class : aistls:WolfPsortInput
  URL: http://www.molprof.jp/ontologies/wolfpsort.rdf#1

・ Blue: required triples to execute a SADI service
e.g. WolfPsort

Subject	Predicate	Object
WolfPsortInput	requiresQueryProteinSequence	string (“animal”, “plant”, or “fungi”)
WolfPsortInput	requiresKingdomInformationOf	string (protein sequence)

The user can download sample input RDFs on http://togo.medals.jp/semantic.eng.html#2 web page (click each links in “Input RDF” column)



SADI service page

A RDF stored execution results of WolfPsort is generated in the following format:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:WolfPsortOutput rdf:about="http://www.molprof.jp/ontologies/wolfpsort.rdf#1">
       <aistls:requiresResultInTextFormat># k used for kNN is: 32
       queryProtein extr 19, E.R. 4, pero 4, lyso 3, E.R._mito 3, mito_pero 3
       </aistls:requiresResultInTextFormat>
    </aistls:WolfPsortOutput>
</rdf:RDF>
```

Wolfpsort output RDF

- Blue: triples for storing WolfPsort results
  e.g. WolfPsort

Subject	Predicate	Object
WolfPsortOutput	requiresResultInTextFormat	string (WolfPsort results)

* If the user doesn't have cURL software, please visit a cURL web site.

http://curl.haxx.se/

1.1. Blast

1.1.1. Preparing input RDF

Blast input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: BlastInput, rdf:about: an arbitrary string
  <aistls:BlastInput rdf:about=”http://www.molprof.jp/ontologies/blast.rdf#1”>
- Triple for a query sequence:
  Subject: BlastInput
  Predicate: requiresQuerySequence
  Object: string (query sequence)
  <aistls:requiresQuerySequence>query
  sequence</aistls:requiresQuerySequence>
- Triple for Blast program name:
  Subject: BlastInput
  Predicate: requiresBlastProgramName
  Object: string (blastp, blastn, blastx, tblastn, tblastx)
  <aistls:requiresBlastProgramName>blastp</aistls:requiresBlastProgramName
>
- Triple for database:
  Subject: BlastInput
  Predicate: requiresBlastDatabase
  Object: string (SWISS, TREMBL, UNIPROT, PROTEIN, PDB etc.)
  *Please visit a below URL for further database information:
  http://blast.ncbi.nlm.nih.gov/Blast.cgi
  <aistls:requiresBlastDatabase>SWISS</aistls:requiresBlastDatabase>
- Triple for E-vlaue:
  Subject: BlastInput

Predicate: setupEvalueThreshold

Object: double (1.0e-10 etc.)

<aistls:setupEvalueThreshold>1.0e-30</aistls:setupEvalueThreshold>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:BlastInput rdf:about="http://www.molprof.jp/ontologies/blast.rdf#1">
       <aistls:requiresQuerySequence>>test
MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGG
GWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNN
FVHDCVNITIKQHTVTTTTKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVGMANLGCWM
LVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQGGGT
HSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFVHDCVNI
TIKQHTVTTTTKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVGMANLGCWMLVLFVATW
SDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQGGGTHSQWNKPS
KPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFVHDCVNITIKQHTVT
TTTKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVGMANLGCWMLVLFVATWSDLGLCKK
RPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQP</aistls:requiresQuerySequence>
       <aistls:requiresBlastDatabase>SWISS</aistls:requiresBlastDatabase>
       <aistls:requiresBlastProgramName>blastp</aistls:requiresBlastProgramName>
       <aistls:setupEvalueThreshold>1.0e-30</aistls:setupEvalueThreshold>
    </aistls:BlastInput>
</rdf:RDF>
```

Blast input RDF

1.1.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/Blast -o "output RDF file"

1.1.3. Execution result

Blast output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: BlastOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:BlastOutput rdf:about=”http://www.molprof.jp/ontologies/blast.rdf#1”>
- Tiple for Blast results:
  Subject: BlastOutput
  Predicate: requiresResultInTextFormat
  Object: string (Blast result)
  <aistls:requiresResultInTextFormat>Blast results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:BlastOutput rdf:about="http://www.molprof.jp/ontologies/blast.rdf#1">
       <aistls:requiresResultInTextFormat>&lt;?xml version="1.0"?&gt;
       &lt;!DOCTYPE BlastOutput PUBLIC "-//NCBI//NCBI BlastOutput/EN"
"http://www.ncbi.nlm.nih.gov/dtd/NCBI_BlastOutput.dtd"&gt;
       &lt;BlastOutput&gt;
         &lt;BlastOutput_program&gt;blastp&lt;/BlastOutput_program&gt;
         &lt;BlastOutput_version&gt;BLASTP 2.2.28+&lt;/BlastOutput_version&gt;
         &lt;BlastOutput_reference&gt;Stephen F. Altschul, Thomas L. Madden, Alejandro A.
Sch&amp;amp;auml;ffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
&amp;quot;Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs&amp;quot;, Nucleic Acids Res. 25:3389-3402.&lt;/BlastOutput_reference&gt;
         &lt;BlastOutput_db&gt;swissprot&lt;/BlastOutput_db&gt;
         &lt;BlastOutput_query-ID&gt;55685&lt;/BlastOutput_query-ID&gt;
         &lt;BlastOutput_query-def&gt;unnamed protein
product&lt;/BlastOutput_query-def&gt;
         &lt;BlastOutput_query-len&gt;1019&lt;/BlastOutput_query-len&gt;
         &lt;BlastOutput_param&gt;
           &lt;Parameters&gt;
             &lt;Parameters_matrix&gt;BLOSUM62&lt;/Parameters_matrix&gt;
             &lt;Parameters_expect&gt;10&lt;/Parameters_expect&gt;
             &lt;Parameters_gap-open&gt;11&lt;/Parameters_gap-open&gt;
……
……
       </aistls:requiresResultInTextFormat>
    </aistls:BlastOutput>
</rdf:RDF>
```

Blast output RDF

1.2. CentroidFold

1.2.1. Preparing input RDF

Centroid input RDF format is as follows:

- RDF header:
  <rdf:RDF
    xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
    xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: CentroidFoldInput, rdf:about: an arbitrary string
  <aistls:CentroidFoldInput
rdf:about=”http://www.molprof.jp/ontologies/centroidfold.rdf#1”>
- Triple for an RNA sequence:
  Subject: CentroidFoldInput
  Predicate: requiresQueryRNASequence
  Object: string (RNA sequence)
  <aistls:requiresQueryRNASequence>RNA sequence
  </aistls:requiresQueryRNASequence>
- Triple for a ClustalW multiple alignment (the user can specify this instead of an RNA sequence as input):
  Subject: CentroidFoldInput
  Predicate: requiresClustalWMultipleAlignment
  Object: string (ClustalW multiple alignment)
  <aistls:requiresClustalWMultipleAlignment>ClustalW multiple alignment
</aistls:requiresClutalWMultipleAlignment>
- Triple for command options:
  Subject: CentroidFoldInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>-g 4</aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:CentroidFoldInput
      rdf:about="http://www.molprof.jp/ontologies/centroidfold.rdf#1">
      <aistls:requiresQueryRNASequence>>AF291576.1/215-270  RF00381;Antizyme_FSE;
      UGAUGCCCCUCACCCAUCAGUGAAGAUCCCGGGUGGGCGAGGGAACGGAA
      GGGAUC
      >AAVX01164114.1/511-453  RF00381;Antizyme_FSE;
      UGAUGUCCCUCACCCACCCUUGAAGAUCCCAGGUGGGCGAGGGAAUGGUC
      AAAGGGAUC
      >BAAE01262592.1/2445-2390  RF00381;Antizyme_FSE;
      UGAUGCCCCUCACCCACAGCUGAAGAUCCCGGGUGGGCGAGGGACUGUCA
      GGGAUC
      </aistls:requiresQueryRNASequence>
      <aistls:hasOptions>-g 4</aistls:hasOptions>
   </aistls:CentroidFoldInput>
</rdf:RDF>
```

CentroidFold input RDF

1.2.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/CentroidFold -o "output RDF file"

1.2.3. Execution result

Centroid output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: CentroidFoldOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:CentroidFoldOutput
  rdf:about=”http://www.molprof.jp/ontologies/centroidfold.rdf#1”>
- Triple for CentroidFold results:
  Subject: CentroidFoldOutput
  Predicate: requiresResultInTextFormat
  Object: string (CentroidFold results)
  <aistls:requiresResultInTextFormat>CentroidFold results
  </aistls:requiresResultInTextFormat>
- Triple for PNG ->Base64 transformation
  Subject: CentroidFoldOutput
  Predicate: requiresResultInBase64BinaryFormat
  Object: string (CentroidFold results)
  <aistls:requiresResultInBase64BinaryFormat>CentroidFold results
  </aistls:requiresResultInBase64BinaryFormat>
  *if “-noimage” is specified at command options triple, CentroidFold doesn’t generate PNG images (not contained in the output RDF file).

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:CentroidFoldOutput
      rdf:about="http://www.molprof.jp/ontologies/centroidfold.rdf#1">
      <aistls:requiresResultInBase64BinaryFormat>PNG:1
      iVBORw0KGgoAAAANSUhEUgAADrcAAA63CAIAAADdRQiTAACAAElEQVR42uzcW3LtxpFAUU7H32dq
      Cv/0iHoMHhzazSNRFJ8A6pWZtVbgU9dE0b6OSmhHvhwAAAAAAAAAAAAAQC0vfgUAAAAAAAAAAAAA
      UIxKGAAAAAAAAAAAAAACqUQkDAAAAAAAAAAAAQDUqYQAAAAAAAAAAAACoRiUMAAAAAAAAAAAAAANWo
      hAEAAAAAAAAAAACgGpUwAAAAAAAAAAAAAFSjEgYAAAAAAAAAAACAalTCAAAAAAAAAAAAAFCNShgA
      AAAAAAAAAAAqlEJAwAAAAAAAAAAAEA1KmEAAAAAAAAAAAAAqEYlDAAAAAAAAAAAAADVqIQBAAAA
      ……
      AACAiVTCcyrhzLS/EmEAAAAAAACYwtQNAABgrlQLd1uGwvkpgCXCAAAAAAAAMJ7BGwAAwFwWCauE
      30iBJcIAAAAAAAAwktkbAADAdCrhcZVwLYJgfTAAAAAAAAAMYwIHAACwwsJVu41D4YqUwRJhAAAA
      AAAAGMMQDgAAYJG9FwnHV8J16YMlwgAAAAAADCAORwAAMA6KuGoSrg6iTAAAAAAAAAQzSgOAABg
      qR9urmevNvTBAAAAAAAAQBwDOeD/t2vHNAAAAAiA+rf2NYCbD+QAAIArx9cSboowAAAAAAAAMBJS
      jJUFYd3SvAAAAABJRU5ErkJggg==
      </aistls:requiresResultInBase64BinaryFormat>
      <aistls:requiresResultInTextFormat>&gt;AF291576.1/215-270  RF00381;Antizyme_FSE;
      UGAUGCCCCUCACCCAUCAGUGAAGAUCCCGGGUGGGCGAGGGAACGGAAGGGAUC
      ....(.((((((.(((((((.(.......)...))))))).))))))..)..... (g=4,th=0.2,e=-10.39)
      &gt;AAVX01164114.1/511-453  RF00381;Antizyme_FSE;
      UGAUGUCCCUCACCCACCCUUGAAGAUCCCAGGUGGGCGAGGGAAUGGUCAAAGGGAUC
      ((((.(((((((.(((((((..((.......))))))))).)))))))...)))).....(g=4,th=0.2,e=-13.76)
      &gt;BAAE01262592.1/2445-2390  RF00381;Antizyme_FSE;
      UGAUGCCCCUCACCCACAGCUGAAGAUCCCGGGUGGGCGAGGGACUGUCAGGGAUC
      (((((.((((((.((((((..(((.......)))))))).))))))..)))))..... (g=4,th=0.2,e=-12.51)
      </aistls:requiresResultInTextFormat>
   </aistls:CentroidFoldOutput>
</rdf:RDF>
```

CentroidFold output RDF

1.3. ClustalW

1.3.1. Preparing input RDF

ClustalW input RDF format is as follows:

- RDF header:
  <rdf:RDF
    xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
    xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: ClustalWInput, rdf:about: an arbitrary string
  <aistls:ClustalWInput
rdf:about=”http://www.molprof.jp/ontologies/clustalw.rdf#1”>
- Triple for multiple sequences
  Subject: ClustalWInput
  Predicate: requiresSequence
  Object: string (DNA or RNA or protein multiple sequences)
  <aistls:requiresSequence>multiple sequences (at least more than two sequences)</aistls:requiresSequence>
- Triple for command options
  Subject: ClustalWInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>-GAPOPEN=10 -GAPEXT=0.5</aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:ClustalWInput rdf:about="http://www.molprof.jp/ontologies/clustalw.rdf#1">
        <aistls:requiresSequence>>1LYLA
        FNDELRNRREKLAALRQQGVAFPNDFRRDHTSDQLHEEFDAKDNQELESLNIEVSVAGRMMTRRIMGKASFVTLQDVGGRI
QLYVARDSLPEGVYNDQFKKWDLGDIIGARGTLFKTQTGELSIHCTELRLLTKALRPLPDQEVRYRQRYLDLIANDKSRQTFVVRSK
ILAAIRQFMVARGFMEVETPMMQVIPGGASARPFITHHNALDLDMYLRIAPELYLKRLVVGGFERVFEINRNFRNEGISVRHNPEFT
MMELYMAYADYHDLIELTESLFRTLAQEVLGTTKVTYGEHVFDFGKPFEKLTMREAIKKYRPETDMADLDNFDAAKALAESIGITVE
KSWGLGRIVTEIFDEVAEAHLIQPTFITEYPAEVSPLARRNDVNPEITDRFEFFIGGREIGNGFSELNDAEDQAERFQEQVNAKAAG
DDEAMFYDEDYVTALEYGLPPTAGLGIGIDRMIMLFTNSHTIRDVILFPAMRP
        >1B8AA
        MYRTHYSSEITEELNGQKVKVAGWVWEVKDLGGIKFLWIRDRDGIVQITAPKKKVDPELFKLIPKLRSEDVVAVEGVVNFT
PKAKLGFEILPEKIVVLNRAETPLPLDPTGKVKAELDTRLNNRFMDLRRPEVMAIFKIRSSVFKAVRDFFHENGFIEIHTPKIIATA
TEGGTELFPMKYFEEDAFLAESPQLYKEIMMASGLDRVYEIAPIFRAEEHNTTRHLNEAWSIDSEMAFIEDEEEVMSFLERLVAHAI
NYVREHNAKELDILNFELEEPKLPFPRVSYDKALEILGDLGKEIPWGEDIDTEGERLLGKYMMENENAPLYFLYQYPSEAKPFYIMK
YDNKPEICRAFDLEYRGVEISSGGQREHRHDILVEQIKEKGLNPESFEFYLKAFRYGMPPHGGFGLGAERLIKQMLDLPNIREVILF
PRDRRRLTP
        ……</aistls:requiresSequence>
        <aistls:hasOptions>-GAPOPEN=10 -GAPEXT=2</aistls:hasOptions>
    </aistls:ClustalWInput>
</rdf:RDF>
```

ClustalW input RDF

1.3.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/ClustalW -o "output RDF file"

1.3.3. Execution result

ClustalW output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: ClustalWOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:ClustalWOutput
  rdf:about=”http://www.molprof.jp/ontologies/clustalw.rdf#1”>
- Triple for ClustalW results
  Subject: ClustalWOutput
  Predicate: requiresResultInTextFormat
  Object: string (ClustalW results)
  <aistls:requiresResultInTextFormat>ClustalW results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:ClustalWOutput rdf:about="http://www.molprof.jp/ontologies/clustalw.rdf#1">
      <aistls:requiresResultInTextFormat>CLUSTAL W (1.83) Multiple Sequence Alignments
      ……
      1ATIA           -WTPPRYFNMMFQDLRGPRGGRGLLAYLRPETAQGIFVNFKNVLDATSRKLGFGIAQIGK
      E64328          ELGEVKKFNLMFVTSIGPGGKR--TGYMRPETAQGIFIQFRRLAQFFRNKLPFGVVQIGK
      B64744          KFRDEVRPRFGVMRSREFLMKDAYSFHTSQESLQETYDAMYAAYSKIFSRMGLDFRAVQA
      E64454          ----------------HGGKTQLDVKLALRPTSETPIYYMMK-LWVKVHTDLPIKIYQIVN
      G64424          -------------------DHGGREMALRPEMTSPVVRFYLNELKNLQKPL--RLYYFAN
      1ADJA           -------------------DRGGRSLTLRPEGTAAMVRAYLEHGMKVWPQP-VRLWMAGP
      G64930          VDMCRGPHVPNMRFCHHFKLMKTAGAYWRGDSNNKMLQRIYGTAWADKKALNAYLQRLEE
      D64449          KGHPLSELSRKIVAKEEKKEEGEESKFYLLNPETEEIIELNENNINIIKDEELLALAKHE
      1SESA           -------------------KALGEEAKRLEEALREKEARLEALLLQVPLPPWPGAPVGGE
      A26400          --------------------KLGEELDAAKAELDALQAEIRDIALTIPNLPADEVPVGKD
      1PYSA           -------------------------EGFRLEGPLGEEVEGRLLLRTHTSPMQVRYMVAHTP
      y|Pyrococcus     --------------LKAIVGVLRKEGWAEVSKTKEGLTLKLSEKGKKAEKRAIDIALEVL
      JT0942          -------------------RPEMAQRLKTRAKITSLVRRFMDDHGFLDIETPMLTKATPE
      1B8AA           ----------------------FTPKAKLGFEILPEKIVVLNRAETPLPLDPTGKVKAELD
      1ASZB           -------------IVKKVDEPIKSATVQNLEIHITKIYTISETPEALPILLEDASRSEAE
      1LYLA           ----------------DSLPEGVYNDQFKKWDLGDIIGARGTLFKTQTGELSIHCTELRL
      S56383          ---------------------FETRFVGPGHSQGMNLWLMTSPEYHMKRLLVAGCGPVFQ
      ……
      ……
      (
      D64449:0.41624,
      G64930:0.40428)
      :0.01577)
      :0.01909);</aistls:requiresResultInTextFormat>
   </aistls:ClustalWOutput>
</rdf:RDF>
```

ClustalW output RDF

1.4. IPknot

1.4.1. Preparing input RDF

IPknot input RDF format is as follows:

- RDF header:
  <rdf:RDF
    xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
    xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: IPknotInput, rdf:about: an arbitrary string
  <aistls:IPknotInput rdf:about=”http://www.molprof.jp/ontologies/ipknot.rdf#1”>
- Triple for an RNA sequence
  Subject: IPknotInput
  Predicate: requiresQueryRNASequence
  Object: string (RNA sequence)
  <aistls:requiresQueryRNASequence>RNA sequence
  </aistls:requiresQueryRNASequence>
- Triple for ClustalW multiple alignment (the user can specify this instead of an RNA sequence as input):
  Subject: IPknotInput
  Predicate: requiresClustalWMultipleAlignment
  Object: string (ClustalW multiple alignment)
  <aistls:requiresClustalWMultipleAlignment>ClustalW multiple alignment
</aistls:requiresClutalWMultipleAlignment>
- Triple for command options:
  Subject: IPknotInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>-i </aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:IPknotInput rdf:about="http://www.molprof.jp/ontologies/ipknot.rdf#1">
         <aistls:requiresQueryRNASequence>>Tomato_mosaic_virus.1
         GUGUCUUGGAGCGCGCGGAGUAAACAUAUAUGGUUCAUAUAUGUCCGUAGGCACGUAAAAAAAGCGA
         >Tobacco_mosaic_virus.1
         GUGUCUUGGAUCGCGCGGGUCAAAUGUAUAUGGUUCAUAUACAUCCGCAGGCACGUAAUAAA-GCGA
         >Rehmannia_mosaic_vir.1
         GUGUCUUGGUUCGCGCGGGUCAAGUGUAUAUGGUGCAUAUACAUCCGUAGGCACGUAAUAAA-GCGA
         >B.pepper.1
         GUGUCUUGGAACGCGCGGGUCAAAUAUAAGUGGUUCACUUAUAUCCGUAGGCACGAAAAAUU-GCGU</aistls:req
uiresQueryRNASequence>
         <aistls:hasOptions>-i</aistls:hasOptions>
    </aistls:IPknotInput>
</rdf:RDF>
```

IPknot input RDF

1.4.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/IPknot -o "output RDF file"

1.4.3. Execution result

IPknot output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: IPknotOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:IPknotOutput
  rdf:about=”http://www.molprof.jp/ontologies/ipknot.rdf#1”>
- Triple for IPknot results:
  Subject: IPknotOutput
  Predicate: requiresResultInTextFormat
  Object: string (IPknot results)
  <aistls:requiresResultInTextFormat>IPknot results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:IPknotOutput rdf:about="http://www.molprof.jp/ontologies/ipknot.rdf#1">
      <aistls:requiresResultInTextFormat>&gt;Tomato_mosaic_virus.1
      GUGUCUUGGAGCGCGCGGAGUAAACAUAUAUGGUUCAUAUAUGUCCGUAGGCACGUAAAAAAAGCGA
      ((((((........((((.....(((((((((...))))))))))))))))).............
      &gt;Rehmannia_mosaic_vir.1
      GUGUCUUGGUUCGCGCGGGUCAAGUGUAUAUGGUGCAUAUACAUCCGUAGGCACGUAAUAAA-GCGA
      ((((((..(((....[[[[)))...(((((((((...)))))))).]]]].))))))............
      ……</aistls:requiresResultInTextFormat>
   </aistls:IPknotOutput>
</rdf:RDF>
```

IPknot output RDF

1.5. Mafft

1.5.1. Preparing input RDF

Mafft input RDF format is as follows:

- RDF header
  <rdf:RDF
    xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
    xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: MafftInput, rdf:about: an arbitrary string
  <aistls:MafftInput rdf:about=”http://www.molprof.jp/ontologies/mafft.rdf#1”>
- Triple for multiple sequences:
  Subject: MafftInput
  Predicate: requiresSequence
  Object: string (DNA or RNA or protein sequences)
  <aistls:requiresSequence>multiple sequences (at least more than two sequences)</aistls:requiresSequence>
- Triple for command options:
  Subject: MafftInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>--retree 2 --maxiterate 0 --bl 62 --op 1.53 --ep 0.0 --clustalout</aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:MafftInput rdf:about="http://www.molprof.jp/ontologies/mafft.rdf#1">
      <aistls:requiresSequence>>1LYLA
      FNDELRNRREKLAALRQQGVAFPNDFRRDHTSDQLHEEFDAKDNQELESLNIEVSVAGRM
      MTRRIMGKASFVTLQDVGGRIQLYVARDSLPEGVYNDQFKKWDLGDIIGARGTLFKTQTG
      ELSIHCTELRLLTKALRPLPDQEVRYRQRYLDLIANDKSRQTFVVRSKILAAIRQFMVAR
      EQVNAKAAGDDEAMFYDEDYVTALEYGLPPTAGLGIGIDRMIMLFTNSHTIRDVILFPAM
      RP
      >1B8AA
      MYRTHYSSEITEELNGQKVKVAGWVWEVKDLGGIKFLWIRDRDGIVQITAPKKKVDPELF
      KLIPKLRSEDVVAVEGVVNFTPKAKLGFEILPEKIVVLNRAETPLPLDPTGKVKAELDTR
      LNNRFMDLRRPEVMAIFKIRSSVFKAVRDFFHENGFIEIHTPKIIATATEGGTELFPMKY
      PNIREVILFPRDRRRLTP
      >1ASZB
      EDTAKDNYGKLPLIQSRDSDRTGQKRVKFVDLDEAKDSDKEVLFRARVHNTRQQGATLAF
      LTLRQQASLIQGLVKANKEGTISKNMVKWAGSLNLESIVLVRGIVKKVDEPIKSATVQNL
      EIHITKIYTISETPEALPILLEDASRSEAEAEAAGLPVVNLDTRLDYRVIDLRTVTNQAI
      FRIQAGVCELFREYLATKKFTEVHTPKLLGAPSEGGSSVFEVTYFKGKAYLAQSPQFNKQ
      QLIVADFERVYEIGPVFRAENSNTHRHMTEFTGLDMEMAFEEHYHEVLDTLSELFVFIFS
      KFLGKLVRDKYDTDFYILDKFPLEIRPFYTMPDPANPKYSNSYDFFMRGEEILSGAQRIH
      ……
      ……
      ……
      EIYEKLKGKFRVHIDDRDIRPGRKFNDWEIKGVPLRIEVGPKDIENKKITLFRRDTMEKF
      QVDETQLMEVVEKTLNNIMENIKNRAWEKFENFITILEDINPDEIKNILSEKRGVILVPF
      KEEIYNEELEEKVEATILGETEYKGNKYIAIAKTY
      </aistls:requiresSequence>
      <aistls:hasOptions></aistls:hasOptions>
   </aistls:MafftInput>
</rdf:RDF>
```

Mafft input RDF

1.5.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/Mafft -o "output RDF file"

1.5.3. Execution result

Mafft output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:aistls "http://www.molprof.jp/ontologies/aistlssio.owl#">
- Subject: MafftOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:MafftOutput rdf:about="http://www.molprof.jp/ontologies/mafft.rdf#1">
- Triple for Mafft results:
  Subject: MafftOutput
  Predicate: requiresResultInTextFormat
  Object: string (Mafft results)
  <aistls:requiresResultInTextFormat>Mafft results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:MafftOutput rdf:about="http://www.molprof.jp/ontologies/mafft.rdf#1">
      <aistls:requiresResultInTextFormat>CLUSTAL format alignment by MAFFT FFT-NS-2
      (v6.717b)
      1LYLA          FNDELRNRREKLAALRQQGVAFPNDFRRDHTSDQLHEEFDAKDNQELESLNIEVSVAGRM
      1B8AA          ------------------------------MYRTHY------SSEITEELNGQKVKVAGWV
      1ASZB          ----------------------------EDTAKDNYGKLPLIQSRDSDRTGQKRVKFVD-L
      1ADJA          ------------------------------------------------------------
      1PYSA          ------------------------------------------------------------


      1LYLA          ------------------------------LVVGGFER------VFEINR-NFRNE-----
      1B8AA          ------------------------------MMASGLDR------VYEIAP-IFRAE-----
      1ASZB          ------------------------------LIVADFER------VYEIGP-VFRAE-----
      1ADJA          -------------------------YLEHGMKVWPQ-----PVRLWMAGP-MFRAE-----
      1PYSA          ------------------------------MVAHTP-----PFRIVVPGR-VFRFE-----
                                                                              *

      ……

      ……


      1LYLA          -----AMRP------------------------------
      1B8AA          -RDRRRLTP------------------------------
      1ASZB          -RDPKRLRP------------------------------
      1ADJA          FLGEDELRAGEVTLKRLATGEQVRLSREEVPGYLLQALG
      1PYSA          ----KFLEQFKGVL-------------------------
      </aistls:requiresResultInTextFormat>
   </aistls:MafftOutput>
</rdf:RDF>
```

Mafft output RDF

1.6. Psipred

1.6.1. Preparing input RDF

Psipred input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PsiPredInput, rdf:about: an arbitrary string
  <aistls:PsiPredInput
  rdf:about=”http://www.molprof.jp/ontologies/psipred.rdf#1”>
- Triple for a protein sequence:
  Subject: PsiPredInput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>
  protein sequence
  </aistls:requiresQueryProteinSequence>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PsiPredInput rdf:about="http://www.molprof.jp/ontologies/psipred.rdf#1">
        <aistls:requiresQueryProteinSequence>>test
        MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGW
GQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDE
YSNQNNFVHDCVNITIKQHTVTTTTKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVGMA
NLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQPHGGGW
GQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFV
HDCVNIMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFVHDCVNITIKQHTVTTT
TKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVG</aistls:requiresQueryProt
einSequence>
    </aistls:PsiPredInput>
</rdf:RDF>
```

Psipred input RDF

1.6.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/Psipred -o "output RDF file"

1.6.3. Execution result

Psipred output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PsiPredOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:PsiPredOutput
  rdf:about=”http://www.molprof.jp/ontologies/psipred.rdf#1”>
- Triple for Psipred results:
  Subject: PsiPredOutput
  Predicate: requiresResultInTextFormat
  Object: string (Psipred results)
  <aistls:requiresResultInTextFormat>Psipred results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:PsiPredOutput rdf:about="http://www.molprof.jp/ontologies/psipred.rdf#1">
      <aistls:requiresResultInTextFormat># PSIPRED HFORMAT (PSIPRED V2.5 by David Jones)
      Conf: 976304410010420000111113889987678876789888999888879888999988
      Pred: CCCCCHHHHHHHHHHHHHCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        AA: MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQP
                  10        20        30        40        50        60



      Conf: 888998988889989888899898888767788735434688777333143200001124
      Pred: CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCHHHHCCCHHHHH
        AA: HGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGA
                  70        80        90       100       110       120



      Conf: 5410003543223207321047743024566655647631254112332854540211 00
      Pred: HHHCCHHHHHHHHHCCCEECCCCCCCCHHHHHHHHHHCCCCCEECCCHHHCCCCCCEEEEE
        AA: VVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFVHDCV
                 130       140       150       160       170       180
      ……
      ……
      Conf: 3899999998764234432234654228605885697303222567721109
      Pred: HHHHHHHHHHHHHHHHHHHHHHHHHHCCCEEEEECCCCHHHHHHHHHHCCC
        AA: DVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVG
                 970       980       990       1000      1010
      </aistls:requiresResultInTextFormat>
   </aistls:PsiPredOutput>
</rdf:RDF>
```

Psipred output RDF

1.7. Raccess

1.7.1. Preparing input RDF

Raccess input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RaccessInput, rdf:about: an arbitrary string
  <aistls:RaccessInput
  rdf:about=”http://www.molprof.jp/ontologies/raccess.rdf#1”>
- Triple for an RNA sequence:
  Subject: RaccessInput
  Predicate: requiresQueryRNASequence
  Object: string (RNA sequence)
  <aistls:requiresQueryRNASequence>RNA sequence
  </aistls:requiresQueryRNASequence>
- Triple for command options:
  Subject: RaccessInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>-access_len=50 </aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:RaccessInput rdf:about="http://www.molprof.jp/ontologies/raccess.rdf#1">
      <aistls:requiresQueryRNASequence>>gi|187607315|ref|NM_014909.4| Homo sapiens
vasohibin 1 (VASH1), mRNA
      GCCCCTGCGCGCCGCCCGAGCCGGTCCCGCTGAGCCGCGGGCCCCGTGCCCTGCGATGGCTCGGCTGGTG
      CAGCGCGGCGCCAGGTGCCAGCCGTCCTCCCGCTGAGACGCGCCCGAGTGGGGACCCGCTGGGCCTCGGG
      GCTCGCAGCCTTCGCCTCCCCGCCGCGCCCGCTCCCTTTCTGGGGACTCCGCCGCTGTTTCTGGGGACGA
      GGGGACAGGGGACCCAGACAAAGCCCACTTTGTGCAGGGAGTTGGCCGCAGGCGGGGAATGTGCGCGTCG
      GCGCGCGCCCCCTCCCCGCTCCCGGCCAGCTGCGAGTCTTGGCTCCCGGACTTGTCTCGTCGCGTCGGAG
      AAATCGCCCCCCAGCGCCGCTCTCCCGCCCGGGGGTCTTGGTTCCGAGCTCGCGCGGCCGGGAGTCGCCT
      CGGTCTTCCTTGGGGCGCGCGCAGATGTGAGCGTGCGAGAGTTGTGTAGGGGATTTTGTTCCCTCCGAAA
      CTGAGACCCAGGGCGCCCAGTGGGCACCCGTGCCTTGACTCTGTCCTTTCTGCAGCCGCTGGTCCGAGCT
      GTCTGGCCTCAGTTTCCCTCCGACTTTTCTCCGCTCTGCCAGCCCTCACTGCTGCCCGTCATTGTTCTCG
      CAGTTAGATGGGGGTGCTTTGTGACGGCTGCCAAGTTGGGGTGTGTTCTCTTTATTCCGTTTTTCAAACA
      GAACAAGGCCTCCAAGGCTGACCCCAGACAACCCACCCCCTCGGACCCTAATTCACCTTATTGCACTGAT
      TTTTTTTATCAAGTCGTATTTTATTGTACAGGAGCCACGCCCTGATTTCTTAAAGGCGCCTTGCACTCTG
      GCCATGTGTTATCTCTGCAGCCGGTGTGTGGGAGGCCTCTTGTGAGCCAGTTGTTTTCCCGCCTCCACCA
      CCCCCCTCGAAGATTTAGGGATGCCAGGGGGGAAGAAGGTGGCTGGGGGTGGCAGCAGCGGTGCCACTCC
      AACGTCCGCTGCGGCCACCGCCCCCTCTGGGGTCAGGCGTTTGGAGACCAGCGAAGGAACCTCAGCCCAG
      CCAAGCTGCTCTCGCTCCCACTGAGCCAAGCCCCCTAACTTTGGGCCTAGAGGCCGTTAGTAT…….
      </aistls:requiresQueryRNASequence>
      <aistls:hasOptions></aistls:hasOptions>
    </aistls:RaccessInput>
</rdf:RDF>
```

Raccess input RDF

1.7.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/Raccess -o "output RDF file"

1.7.3. Execution result

Raccess output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RaccessOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:RaccessOutput
  rdf:about=”http://www.molprof.jp/ontologies/raccess.rdf#1”>
- Triple for Raccess results:
  Subject: RaccessOutput
  Predicate: requiresResultInTextFormat
  Object: string (Raccess results)
  <aistls:requiresResultInTextFormat>Raccess results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:RaccessOutput rdf:about="http://www.molprof.jp/ontologies/raccess.rdf#1">
      <aistls:requiresResultInTextFormat>&gt;gi|187607315|ref|NM_014909.4| Homo sapiens
vasohibin 1 (VASH1), mRNA
      6020    6029    0.62373
      6019    6028    0.623859
      6018    6027    0.623156
      6017    6026    0.622921
      6016    6025    0.648241
      6015    6024    1.34601
      6014    6023    2.08364
      6001    6010    9.45949
      6000    6009    8.523
      5999    6008    9.24843
      …….
      …….
      11     20     11.5702
      10     19     11.5887
      9      18     11.4777
      8      17     11.4848
      7      16     13.0533
      6      15     12.7788
      5      14     10.5224
      4      13     11.3123
      3      12     10.8754
      2      11     10.6341
      1      10     9.07663
      0      9      6.90453/aistls:requiresResultInTextFormat>
   </aistls:RaccessOutput>
</rdf:RDF>
```

Raccess output RDF

1.8. RactIP

1.8.1. Preparing input RDF

RactIP input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RactIPInput, rdf:about: an arbitrary string
  <aistls:RactIPInput rdf:about=”http://www.molprof.jp/ontologies/ractip.rdf#1”>
- Triple for a query RNA sequence:
  Subject: RactIPInput
  Predicate: requiresQueryRNASequence
  Object: string (RNA sequence)
  <aistls:requiresQueryRNASequence>RNA sequence
  </aistls:requiresQueryRNASequence>
- Triple for a target RNA sequence:
  Subject: RactIPInput
  Predicate: requiresSubjectRNASequence
  Object: string (RNA sequence)
  <aistls:requiresSubjectRNASequence>RNA sequence
  </aistls:requiresSubjectRNASequence>
- Triple for command options:
  Subject: RactIPInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions>-i</aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:RactIPInput rdf:about="http://www.molprof.jp/ontologies/ractip.rdf#1">
      <aistls:requiresQueryRNASequence>>R1inv
      GGCAACGGAUGGUUCGUUGCC</aistls:requiresQueryRNASequence>
      <aistls:requiresSubjectRNASequence>>R2inv
      GCACCGAACCAUCCGGUGC</aistls:requiresSubjectRNASequence>
      <aistls:hasOptions>-i</aistls:hasOptions>
   </aistls:RactIPInput>
</rdf:RDF>
```

RactIP input RDF

1.8.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/RactIP -o "output RDF file"

1.8.3. Execution result

RactIP output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RactIPOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:RactIPOutput
  rdf:about=”http://www.molprof.jp/ontologies/ractip.rdf#1”>
- Triple for RactIP results:
  Subject: RactIPOutput
  Predicate: requiresResultInTextFormat
  Object: string (RactIP results)
  <aistls:requiresResultInTextFormat>RactIP results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:RactIPOutput rdf:about="http://www.molprof.jp/ontologies/ractip.rdf#1">
      <aistls:requiresResultInTextFormat>*    0:   objval =   0.000000000e+00   infeas =
0.000000000e+00 (0)
      *    36:   objval =   1.604034567e+01   infeas =   0.000000000e+00 (0)
      OPTIMAL SOLUTION FOUND
      Integer optimization begins...
      +    36: mip =     not found yet &lt;=            +inf       (1; 0)
      +    36: &gt;&gt;&gt;&gt;&gt;   1.604034567e+01 &lt;=   1.604034567e+01   0.0% (1; 0)
      +    36: mip =   1.604034567e+01 &lt;=    tree is empty   0.0% (0; 1)
      INTEGER OPTIMAL SOLUTION FOUND
      &gt;R1inv
      GGCAACGGAUGGUUCGUUGCC
      (((((((([[[[[[]))))))))
      &gt;R2inv
      GCACCGAACCAUCCGGUGC
      ((((((]]]]]]]]]]))))))</aistls:requiresResultInTextFormat>
   </aistls:RactIPOutput>
</rdf:RDF>
```

RactIP output RDF

1.9. Wolfpsort

1.9.1. Preparing input RDF

Wolfpsort input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: WolfPsortInput, rdf:about: an arbitrary string
  <aistls:WolfPsortInput
  rdf:about=”http://www.molprof.jp/ontologies/wolfpsort.rdf#1”>
- Triple for kingdom information:
  Subject: WolfPsortInput
  Predicate: requiresKingdomInformation
  Object: string (“animal” or “plant” or “fungi”)
  <aistls:requireKingomdInfomationOf>animal
  </aistls:requireKingdomInformationOf>
- Triple for a protein sequence:
  Subject: WolfPsortInput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>
  protein sequence
  </aistls:requiresQueryProteinSequence>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:WolfPsortInput rdf:about="http://www.molprof.jp/ontologies/wolfpsort.rdf#1">
      <aistls:requiresKingdomInformationOf>animal</aistls:requiresKingdomInformationOf>
      <aistls:requiresQueryProteinSequence>MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSP
GGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLG
SAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDEYSNQNNFVHDCVNITIKQHTVTTTTKGENFTETDVKMMERVVEQMCITQY
ERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVG</aistls:requiresQueryProteinSequence>
   </aistls:WolfPsortInput>
</rdf:RDF>
```

Wolfpsort input RDF

1.9.2. Execution command

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/Wolfpsort -o "output RDF file"

1.9.3. Execution result

WolfPsort output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: WolfPsortOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:WolfPsortOutput
  rdf:about=”http://www.molprof.jp/ontologies/wolfpsort.rdf#1”>
- Triple for Wolfpsort results:
  Subject: WolfPsortOutput
  Predicate: requiresResultInTextFormat
  Object: string (Wolfpsort results)
  <aistls:requiresResultInTextFormat>Wolfpsort results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:WolfPsortOutput rdf:about="http://www.molprof.jp/ontologies/wolfpsort.rdf#1">
      <aistls:requiresResultInTextFormat># k used for kNN is: 32
      queryProtein extr 19, E.R. 4, pero 4, lyso 3, E.R._mito 3, mito_pero 3
      </aistls:requiresResultInTextFormat>
   </aistls:WolfPsortOutput>
</rdf:RDF>
```

Wolfpsort output RDF

2. Asynchronous type SADI services

2.0. Uses asynchronous type SADI services

Asynchronous type SADI services are Last, Modelling, PoodleL, PoodleS and Rassie.

Each SADI service is executed by the following three procedures:

1） Getting polling URL

% curl --data-binary @"input RDF file"
http://semantic.medals.jp:8090/sadi-services/"SADI service name" (Figure 1-A)

If the user would like to execute PoodleL SADI service, the user specify below command:

% curl --data-binary @input.rdf
http://semantic.medals.jp:8090/sadi-services/PoodleL

An RDF-format stored a polling URL is displayed in command lines as below.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PoodleLOutput rdf:about="http://www.molprof.jp/ontologies/poodle1.rdf#1">
        <rdfs:isDefinedBy
        rdf:resource="http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=1"/>
    </aistls:PoodleLOutput>
</rdf:RDF>
```

RDF stored a polling URL (e.g. PoodleL)

2） Polling to SADI server

% curl http://semantic.medals.jp:8090/sadi-services/”SADI service name”?poll=”random polling number”

If the user asks to the SADI server whether PoodleL execution is completed or not:

%curl http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=1

If service is completed, a URL to the RDF file stored the results is displayed in command lines as below:

% curl http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=1
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/poodleLResult.rdf
*This result is displayed in command lines one time.

3） Getting execution results

% curl http://semantic.medals.jp/tmp/xxx/yyyy/ poodleLResult.rdf -o ”output RDF file”

Please visit a chapter 1.0 for further input and output RDF format information.

2.1. Last

2.1.1. Preparing input RDF

Last input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: LastInput, rdf:about: an arbitrary string
  <aistls:LastInput rdf:about=”http://www.molprof.jp/ontologies/last.rdf#1”>
- Triple for a query sequence:
  Subject: LastInput
  Predicate: requiresQuerySequence
  Object: string (query sequence)
  <aistls:requiresQuerySequence>query
  sequence</aistls:requiresQuerySequence>
- Triple for a target sequence:
  Subject: LastInput
  Predicate: requiresSubjectSequence
  Object: string (target sequence)
  <aistls:requiresSubjectSequence>target sequence
  </aistls:requiresSubjectSequence>
- Triple for command options for Lastdb:
  Subject: LastInput
  Predicate: hasOptionsForLastdb
  Object: string (command options for Lastdb)
  <aistls:hasOptionsForLastdb>-m110 -w1</aistls:hasOptionsForLastdb>
- Triple for command options for Lastal:
  Subject: LastInput
  Predicate: hasOptionsForLastal
  Object: string (command options for Lastal)

<aistls:hasOptionsForLastal>-j4 -u0 -m10 -l1 -k1 -w0 -g1.0 -s2 -e30
</aistls:hasOptionsForLastal>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:LastInput rdf:about="http://www.molprof.jp/ontologies/last.rdf#1">
      <aistls:requiresQuerySequence>>gi|14518450|ref|NC_000868.1| Pyrococcus abyssi GE5,
complete genome
      GGGCTTTAGCCTCCTTCACCGCTTCCACGATTTTCTGCCTGTCAAAGGGCATTCTAGACATCCCTCCTTA
      GGTTTTTAATTAAAAATTCAAGGTGGAGTAAAAAGGGATGTTTTTAAATTTTTCTCACTCTTTCTCGGCC
      TTCTCAAATAGCTCGTCGTAAACCCCTTCATCTATTTCTCTCTGAACTTCCCTTGGATCCTTGCCTTCGA
      CGGTAACTCCCATGCTTAAAGCCGTTCCAATGACTTCCTTGGCGGCAGCCTTAAGAGTCAATGCTAGCAT
      CTGGTTTCTCTTCATCTTAGCTATCTTGATAACTTGCTCCATCGTTAAGTTCCCAACGATATTGTGCTTC
      CTATCTCGAACTGCTTGGTTACTGGATCTACGATGATCTTCACTGGGACCTGCATCCCAGCGAACTCTTT
      ……
      AACATCAAGAAGCTGACCTACCACGGCCCTGAACTTCCTAGGATCTCCATGTCCCTCATCCTCTTCTTCA</aistls:re
quiresQuerySequence>
      <aistls:requiresSubjectSequence>
      AGATCCTTAGCCTTGTTGTTCCTTTTCTCAAGGAGCTTTACGCTACCGTCTTCACAGATCTCATAGATCG
      CGAAAAACTCTGAATCTCCGTAGTGAGCGTCTATGAGATGTTCATCATCCTCCATTCCAAAGGCTACCTT
      ……
      GTTCCTGGGACCTAGGTATCTACCGAGGTACCTACCTATCTTGGGCATTAATGGGGCCTCAGCTATGAAG
      TAATAACGTCAAGCCCGAGCCTCCTCGCCGCTTCGGCAACTGCACCATCAGCGATGACCGCGATCTTTAC
      TCTTTAAGGTTCACTGCCACCTCGACACTCTGTGTGAAGTTACGCGGCTTGGCCC</aistls:requiresSubjectSe
quence>
      <aistls:hasOptionsForLastdb>-m110 -w1</aistls:hasOptionsForLastdb>
      <aistls:hasOptionsForLastal>-j4 -u0 -m10 -l1 -k1 -w0 -g1.0 -s2
-e30</aistls:hasOptionsForLastal>
   </aistls:LastInput>
</rdf:RDF>
```

Last input RDF

2.1.2. Execution command

% curl --data-binary @"input RDF"
http://semantic.medals.jp:8090/sadi-services/Last

Last is an asynchronous SADI service, and first an RDF stored a URL polling for the SADI server is displayed in command lines as below. Next, the user asks to the SADI server using URL defined in "isDefinedBy" tag whether Last execution is completed or not.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:LastOutput rdf:about="http://www.molprof.jp/ontologies/last.rdf#1">
        <rdfs:isDefinedBy
rdf:resource="http://semantic.medals.jp:8090/sadi-services/Last?poll=8"/>
        </aistls:LastOutput>
</rdf:RDF>
```

RDF stored the URL to poll for the SADI server

% curl http://semantic.medals.jp:8090/sadi-services/Last?poll=8 (in this case)
%

If Last execution is completed, the URL of an output RDF file stored Last results is displayed in command lines.
% curl http://semantic.medals.jp:8090/sadi-services/Last?poll=8
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/seq1_seq2Result.rdf
%

The user can get the RDF file as follows:
% curl http://semantic.medals.jp/tmp/xxx/yyyy/ seq1_seq2Result.rdf -o "output name"

2.1.3. Execution result

Last output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: LastOutput, rdf:about: the string as same as specifying in the input RDF file
  <aistls:LastOutput rdf:about=”http://www.molprof.jp/ontologies/last.rdf#1”>
- Triple for Last results:
  Subject: LastOutput
  Predicate: requiresResultInTextFormat
  Object: string (Last results)
  <aistls:requiresResultInTextFormat>Last results
  </aistls:requiresResultInTextFormat>
- Triple for PNG ->Base64 transformation:
  Subject: LastOutput
  Predicate: requiresResultInBase64BinaryFormat
  Object: string (Last results)
  <aistls:requiresResultInBase64BinaryFormat>Last results
  </aistls:requiresResultInBase64BinaryFormat>
  *if “-noimage” is specified at command options triple, Last doesn’t generate PNG images (not contained in the output RDF file).

Sample output RDF file is as follows:


```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:LastOutput rdf:about="http://www.molprof.jp/ontologies/last.rdf#1">
      <aistls:requiresResultInBase64BinaryFormat>iVBORw0KGgoAAAANSUhEUgAAA8cAAAPoCAAAAA
DWntNlAAANoElEQVR42u3dAXIjRRZF0dz/ZmsJ
      oHbLqlLZbURnNc73OCcmGMIw0GP7kvm/RHlsQLrhUwA65k9+dcaH37/91/jh/SPjw8fHGL/6q42v
      /lbjH3/7ya8KHfP61+ie7ltQj98b93xPH79/5Bf/UBjvf/J4/EUf/6uv//MIfvjO0TGvHcRjPBdz
      ……
      ……
      AAAASUVORK5CYII=</aistls:requiresResultInBase64BinaryFormat>
      <aistls:requiresResultInTextFormat># LAST version 58
      #
      # a=7 b=1 c=100000 e=30 d=18 x=27 y=10
      # u=0 s=2 m=10 l=1 k=1 i=134217728 w=0 t=0.910239 g=1 j=4
      # seq1
      ……
      ……

      a score=30
      s gi|14518450|ref|NC_000868.1|                                    29878 36 +
139440GTTGAGAGTTCCAATAAGACTAAAATAGGATTGAAA
      s GATCCTTAGCCTTGTTGTTCCTTTTCTCAAGGAGCTTTACGCTACCGTCTTCACAGATCTCATAGATCG 130267 36 -
144325 GTGGCGAGTTCCAATAAGACTAAAATAGAATTGAAA
      p 0.551 0.673 0.714 0.836 0.878 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0.999
0.999 0.998 0.996 0.99 0.972 0.916 0.749

      # CPU time: 0.36 seconds
      </aistls:requiresResultInTextFormat>
   </aistls:LastOutput>
</rdf:RDF>
```


Last output RDF

2.2. Modelling

2.2.1. Preparing input RDF

Modelling input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: ModellingInput, rdf:about: an arbitrary string
  <aistls:ModellingInput
  rdf:about=”http://www.molprof.jp/ontologies/modelling.rdf#1”>
- Triple for a protein sequence:
  Subject: ModellingInput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>protein sequence
  </aistls:requiresQueryProteinSequence>
- Triple for BLAST program name:
  Subject: ModellingInput
  Predicate: requiresBlastProgramName
  Object: string (“BLAST” or “PSI-BLAST”)
  <aistls:requiresBlastProgramName>PSI-BLAST</aistls:requiresBlastProgram
  Name>
- Triple for iteration number (for PSI-BLAST):
  Subject: ModellingInput
  Predicate: setupIterationNumber
  Object: integer (iteration number >1)
  <aistls:setupIterationNumber>3</aistls:setupIterationNumber>
- Triple for E-Value:
  Subject: ModellingInput
  Predicate: setupEvalueThreshold

Object: double (E-Value)
<aistls:setupEvalueThreshold>0.00001</aistls:setupEvalueThreshold>

- Triple for a coverage threshold for BLAST hit regions:
  Subject: ModellingInput
  Predicate: setupHitRegionCoverageThreshold
  Object: double (coverage threshold for BLAST hit regions)
  <aistls:setupHitRegionCoverageThreshold>60.0
  </aistls:setupHitRegionCoverageThreshold>
- Triple for an identity threshold for BLAST hit regions:
  Subject: ModellingInput
  Predicate: setupHitRegionIdentityThreshold
  Object: double (identity threshold for BLAST hit regions)
  <aistls:setupHitRegionIdentityThreshold>30.0
  </aistls:setupHitRegionIdentityThreshold>
- Triple for a minimum sequence length threshold of BLAST hit regions:
  Subject: ModellingInput
  Predicate: setupMinimumSequenceLength
  Object: integer (minimum sequence length threshold of BLAST hit regions)
  <aistls:setupMinimumSequenceLength>30
  </aistls:setupMinimumSequenceLength>
- Triple for a coverage threshold for selecting template sequences:
  Subject: ModellingInput
  Predicate: setupTemplateCoverageThreshold
  Object: double (coverage threshold for selecting template sequences)
  <aistls:setupTemplateCoverageThreshold>90.0
  </aistls:setupTemplateCoverageThreshold>
- Triple for an identity threshold for selecting template sequences:
  Subject: ModellingInput
  Predicate: setupTemplateIdentityThreshold
  Object: double (identity threshold for selecting template sequences)
  <aistls:setupTemplateIdentityThreshold>90.0
  </aistls:setupTemplateIdentityThreshold>
- Triple for MODELLER license key:
  Subject: ModellingInput
  Predicate: requiresLicenseKey
  Object: string (MODELLER license key)

<aistls:requiresLicenseKey>***</aistls:requiresLicenseKey>
*Modelling execution needs MODELLER license key. Please visit a MODELLER registration site (http://saliab.org/modeller/registration.html).

- Triple for a threshold of number of models generated by MODELLER:
  Subject: ModellingInput
  Predicate: setupModelNumber
  Object: integer (threshold of number of models generated by MODELLER)
  <aistls:setupModelNumber>10</aistls:setupModelNumber>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:ModellingInput rdf:about="http://www.molprof.jp/ontologies/modelling.rdf#1">
      <aistls:requiresQueryProteinSequence>>sample
      MNGTEGPNFYVPFSNKTGVVRSPFEAPQYYLAEPWQFSMLAAYMFLLIMLGFPINFLTLYVTVQHKKLRTPLNYILLNLAV
ADLFMVFGGFTTTLYTSLHGYFVFGPTGCNLEGFFATLGGEIALWSLVVLAIERYVVVCKPMSNFRFGENHAIMGVAFTWVMALACA
APPLVGWSRYIPEGMQCSCGIDYYTPHEETNNESFVIYMFVVHFIIPLIVIFFCYGQLVFTVKEAAAQQQESATTQKAEKEVTRMVI
IMVIAFLICWLPYAGVAFYIFTHQGSDFGPIFMTIPAFFAKTSAVYNPVIYIMMNKQFRNCMVTTLCCGKNKREIRLMKNREAAREC
RRKKKEYVKCLENRVAVLENQNKTLIEELKTLKDLYSNKMSEEGPQVKIREASKDNVDFILSNVDLAMANSLRRVMIAEIPTLAIDS
VEVETNTTVLADEFIAHRLGLIPLQSMDIEQLEYSRDCFCEDHCDKCSVVLTLQAFGESESTTNVYSKDLVIVSNLMGRNIGHPIIQ
DKEGNGVLICKLRKGQELKLTCVAKKGIAKEHAKWGPAAAIEFEYDPWNKLKHTDYWYEQDSAKEWPQSKNCEYEDPPNEGDPFDYK
AQADTFYMNVESVGSIPVDQVVVRGIDTLQKKVASILLALTQMDQDKVNFASGDNNTASNMLGSNEDVMMTGAEQDPYSNASQMGNT
GSGGYDNAW</aistls:requiresQueryProteinSequence>
      <aistls:requiresBlastProgramName>BLAST</aistls:requiresBlastProgramName>
      <aistls:setupEvalueThreshold>0.00005</aistls:setupEvalueThreshold>
      <aistls:setupHitRegionCoverageThreshold>60.0</aistls:setupHitRegionCoverageThresh
old>
      <aistls:setupHitRegionIdentityThreshold>30.0</aistls:setupHitRegionIdentityThresh
old>
      <aistls:setupMinimumSequenceLength>30</aistls:setupMinimumSequenceLength>
      <aistls:setupTemplateCoverageThreshold>95.0</aistls:setupTemplateCoverageThreshol
d>
      <aistls:setupTemplateIdentityThreshold>95.0</aistls:setupTemplateIdentityThreshol
d>
```

Modelling input RDF

2.2.2. Execution command

% curl --data-binary @"input RDF"
http://semantic.medals.jp:8090/sadi-services/Modelling

Modelling is an asynchronous SADI service, and first an RDF stored a URL polling for the SADI server is displayed in command lines as below. Next, the user asks to the SADI server using the URL defined in "isDefinedBy" tag whether Modelling execution is completed or not.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:ModellingOutput rdf:about="http://www.molprof.jp/ontologies/modelling.rdf#1">
        <rdfs:isDefinedBy
rdf:resource="http://semantic.medals.jp:8090/sadi-services/Modelling?poll=10"/>
    </aistls:ModellingOutput>
</rdf:RDF>
```

RDF stored the URL to poll for the SADI server

% curl http://semantic.medals.jp:8090/sadi-services/Modelling?poll=10 (in this case)
%

If Modelling execution is completed, the URL of an output RDF file stored Modelling results is displayed in command lines.
% curl http://semantic.medals.jp:8090/sadi-services/Modelling?poll=10
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/modellingResult.rdf
%

The user can get the RDF file as follows:
% curl http://semantic.medals.jp/tmp/xxx/yyyy/modellingResult.rdf -o "output name"

2.2.3. Execution result

Modelling output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: ModellingOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:ModellingOutput
  rdf:about=”http://www.molprof.jp/ontologies/modelling.rdf#1”>
- Triple for a protein sequence:
  Subject: ModellingOutput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>protein sequence
  </aistls:requiresQueryProteinSequence>
- Triple for BLAST results:
  Subject: ModellingOutput
  Predicate: requiresBlastResult
  Object: string (BLAST results)
  <aistls:requiresBlastResult>BLAST results</aistls:requiresBlastResult>
- Triple for hit region information for modelling:
  Subject: ModellingOutput
  Predicate: requiresHitRegionInformation
  Object: string (hit region information for modelling)
  <aistls:requiresHitRegionInformation>hit region information
  </aistls:requiresHitRegionInformation>
- Triple for MODELLER results:
  Subject: ModellingOutput
  Predicate: requiresModellerResult
  Object: string (MODELLER results)
  <aistls:requiresModellerResult>MODELLER results

</aistlsrequiresModellerResult>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:ModellingOutput rdf:about="http://www.molprof.jp/ontologies/modelling.rdf#1">
      <aistls:requiresBlastResult>
      PDB      QHitRegionLen   QCov    QHitRange      THitRegionLen   Coverage
THitRange      Identity       E-value
      1qm0A   104     41.11   125-228 104     100.00  1-104   100.00  3.20122e-60
      1xyuA   111     43.87   121-231 111     100.00  1-111   90.99   1.43738e-59
      ……</aistls:requiresBlastResult>
      <aistls:requiresHitRegionInformation>PDB     QHitRegionLen   QCov    QHitRange
THitRegionLen   TCov    THitRange      Identity       E-value
      1qm0A   104     41.11   125-228 104     100.00  1-104   100.00  3.20122e-60
      </aistls:requiresHitRegionInformation>
      <aistls:requiresModellerResult>MODEL:1       Query hit region:1-326
      EXPDTA    THEORETICAL MODEL, MODELLER 9v5 2012/02/09 21:51:43
      REMARK   6 MODELLER OBJECTIVE FUNCTIO
      N:     1915.0485
      REMARK   6 MODELLER BEST TEMPLATE % SEQ ID: 100.000
      ATOM    252  ND2 ASN   355     57.353  10.420  32.011  1.00112.53          N
      ATOM    253  C   ASN   355     55.628   8.704  34.015  1.00112.53          C
      ATOM    254  O   ASN   355     54.440   8.414  33.908  1.00112.53          O
      ……
      ……</aistls:requiresModellerResult>
      <aistls:requiresQueryProteinSequence>&gt;sp|P04156|PRIO_HUMAN Major prion protein
OS=Homo sapiens GN=PRNP PE=1 SV=1
    MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQP
    HGGGWGQPHGGGWGQPHGGGWGQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGA</aistls:requiresQueryPr
oteinSequence>
    </aistls:ModellingOutput>
</rdf:RDF>
```

Modelling output RDF

2.3. PoodleL

2.3.1. Preparing input RDF

PoodleL input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PoodleLInput, rdf:about: an arbitrary string
  <aistls:PoodleLInput
  rdf:about=”http://www.molprof.jp/ontologies/poodlel.rdf#1”>
- Triple for a protein sequence:
  Subject: PoodleLInput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>protein sequence
  </aistls:requiresQueryProteinSequence>

Sample input RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:PoodleLInput rdf:about="http://www.molprof.jp/ontologies/poodlel.rdf#1">
      <aistls:requiresQueryProteinSequence>>test
      MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGWGQPHGGGW
GQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDE
YSNQNNFVHDCVNITIKQHTVTTTTKGENFTETDVKMMERVVEQMCITQYERESQAYYQRGSSMVLFSSPPVILLISFLIFLIVGMA
</aistls:requiresQueryProteinSequence>
   </aistls:PoodleLInput>
</rdf:RDF>
```

PoodleL input RDF

2.3.2. Execution command

% curl --data-binary @"input RDF"
http://semantic.medals.jp:8090/sadi-services/PoodleL

PoodleL is an asynchronous SADI service, and first an RDF stored a URL polling for the SADI server is displayed in command lines as below. Next, the user asks to the SADI server using the URL defined in "isDefinedBy" tag whether PoodleL execution is completed or not.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PoodleLOutput rdf:about="http://www.molprof.jp/ontologies/poodlel.rdf#1">
        <rdfs:isDefinedBy
rdf:resource="http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=21"/>
    </aistls:PoodleLOutput>
</rdf:RDF>
```

RDF stored the URL to poll for the SADI server

% curl http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=21 (in this case)
%

If PoodleL execution is completed, the URL of an output RDF file stored PoodleL results is displayed in command lines.
% curl http://semantic.medals.jp:8090/sadi-services/PoodleL?poll=21
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/poodleLResult.rdf
%

The user can get the RDF file as follows:
% curl http://semantic.medals.jp/tmp/xxx/yyyy/ poodleLResult.rdf -o "output name"

2.3.3. Execution result

PoodleL output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PoodleLOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:PoodleLOutput
  rdf:about=”http://www.molprof.jp/ontologies/poodlel.rdf#1”>
- Triple for PoodleL results:
  Subject: PoodleLOutput
  Predicate; requiresResultInTextFormat
  Object: string (PoodleL results)
  <aistls:requiresResultInTextFormat>PoodleL results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PoodleLOutput rdf:about="http://www.molprof.jp/ontologies/poodlel.rdf#1">
        <aistls:requiresResultInTextFormat>PFRMAT DR
        REMARK S. Hirose, K. Shimizu, S. Kanai, Y. Kuroda and T. Noguchi
        …..
        A O 0.1241
        P O 0.2739
        END
        ……</aistls:requiresResultInTextFormat>
    </aistls:PoodleLOutput>
</rdf:RDF>
```

PoodleL output RDF

2.4. PoodleS

2.4.1. Preparing input RDF

PoodleS input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PoodleSInput, rdf:about: an arbitrary string
  <aistls:PoodleSInput
  rdf:about=”http://www.molprof.jp/ontologies/poodles.rdf#1”>
- Triple for a protein sequence:
  Subject: PoodleSInput
  Predicate: requiresQueryProteinSequence
  Object: string (protein sequence)
  <aistls:requiresQueryProteinSequence>protein sequence
  </aistls:requiresQueryProteinSequence>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PoodleSInput rdf:about="http://www.molprof.jp/ontologies/poodles.rdf#1">
       <aistls:requiresQueryProteinSequence>>test
       MANLGCWMLVLFVATWSDLGLCKKRPKPGGWNTGGSRYPGQGSPGGNRYPPQGGGGWGQPHGGGWGQPHGGGW
GQPHGGGWGQGGGTHSQWNKPSKPKTNMKHMAGAAAAGAVVGGLGGYMLGSAMSRPIIHFGSDYEDRYYRENMHRYPNQVYYRPMDE
YSNQNNFVHDCVNITIKQHTVTTTTKGENFTETDVKMMERVVE</aistls:requiresQueryProteinSequence>
    </aistls:PoodleSInput>
</rdf:RDF>
```

PoodleS input RDF

2.4.2. Execution command

% curl --data-binary @"input RDF"
http://semantic.medals.jp:8090/sadi-services/PoodleS

PoodleS is an asynchronous SADI service, and first an RDF stored a URL polling for the SADI server is displayed in command lines as below. Next, the user asks to the SADI server using the URL defined in "isDefinedBy" tag whether PoodleS execution is completed or not.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:PoodleSOutput rdf:about="http://www.molprof.jp/ontologies/poodles.rdf#1">
        <rdfs:isDefinedBy
rdf:resource="http://semantic.medals.jp:8090/sadi-services/PoodleS?poll=3F"/>
    </aistls:PoodleSOutput>
</rdf:RDF>
```

RDF stored the URL to poll for the SADI server

% curl http://semantic.medals.jp:8090/sadi-services/PoodleS?poll=3F (in this case)
%

If PoodleS execution is completed, the URL of an output RDF file stored PoodleS results is displayed in command lines.
% curl http://semantic.medals.jp:8090/sadi-services/PoodleS?poll=3F
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/poodleSResult.rdf
%

The user can get the RDF file as follows:
% curl http://semantic.medals.jp/tmp/xxx/yyyy/ poodleSResult.rdf -o "output name"

2.4.3. Execution result

PoodleS output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: PoodleSOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:PoodleSOutput
  rdf:about=”http://www.molprof.jp/ontologies/poodles.rdf#1”>
- Triple for PoodleS results:
  Subject: PoodleSOutput
  Predicate: requiresResultInTextFormat
  Object: string (PoodleS results)
  <aistls:requiresResultInTextFormat>PoodleS results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:PoodleSOutput rdf:about="http://www.molprof.jp/ontologies/poodles.rdf#1">
      <aistls:requiresResultInTextFormat>PFRMAT DR
      REMARK K. Shimizu, Y. Muraoka, S. Hirose, and T. Noguchi
      REMARK "Feature Selection Based on Physicochemical Properties o
      A D  0.575
      N D 0.505
      …..
      Pharma Design, Inc. </aistls:requiresResultInTextFormat>
   </aistls:PoodleSOutput>
</rdf:RDF>
```

PoodleS output RDF

2.5. Rassie

2.5.1. Preparing input RDF

Rassie input RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RassieInput, rdf:about: an arbitrary string
  <aistls:RassieInput rdf:about=”http://www.molprof.jp/ontologies/rassie.rdf#1”>
- Triple for an RNA secondary structure:
  Subject: RassieInput
  Predicate: requiresRNASecondaryStructure
  Object: string (RNA secondary structure)
  <aistls:requiresRNASecondaryStructure>RNA secondary structure
  </aistls:requiresRNASecondaryStructure>
- Triple for command options:
  Subject: RassieInput
  Predicate: hasOptions
  Object: string (command options)
  <aistls:hasOptions> -q 100 -ins 100 -clst -outclst 10 c-ins_hain
  </aistls:hasOptions>

Sample input RDF file is as follows:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:RassieInput rdf:about="http://www.molprof.jp/ontologies/rassie.rdf#1">
        <aistls:requiresRNASecondaryStructure>>1CQ5
        GGCGUUUACCAGGUCAGGUCCGGAAGGAAGCAGCCAAGGCGCC
        (((((((....(((....(((....))))....))).))))))) (g=4,th=0.2)
        </aistls:requiresRNASecondaryStructure>
        <aistls:hasOptions> -q 100 -ins 100 -clst -outclst 10 -ins_chain</aistls:hasOptions>
    </aistls:RassieInput>
</rdf:RDF>
```

Rassie input RDF

2.5.2. Execution command

% curl --data-binary @"input RDF"
http://semantic.medals.jp:8090/sadi-services/Rassie

Rassie is an asynchronous SADI service, and first an RDF stored a URL polling for the SADI server is displayed in command lines as below. Next, the user asks to the SADI server using the URL defined in "isDefinedBy" tag whether Rassie execution is completed or not.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
    <aistls:RassieOutput rdf:about="http://www.molprof.jp/ontologies/rassie.rdf#1">
       <rdfs:isDefinedBy
rdf:resource="http://semantic.medals.jp:8090/sadi-services/Rassie?poll=30"/>
    </aistls:RassieOutput>
</rdf:RDF>
```

RDF stored the URL to poll for the SADI server

% curl http://semantic.medals.jp:8090/sadi-services/Rassie?poll=30 (in this case)
%

If Rassie execution is completed, the URL of an output RDF file stored Rassie results is displayed in command lines.
% curl http://semantic.medals.jp:8090/sadi-services/Rassie?poll=30
COMPLETE: http://semantic.medals.jp/tmp/xxx/yyyy/rna3dResult.rdf
%

The user can get the RDF file as follows:
% curl http://semantic.medals.jp/tmp/xxx/yyyy/rna3dResult.rdf -o "output name"

2.5.3. Execution result

Rassie output RDF format is as follows:

- RDF header:
  <rdf:RDF
  xmlns:rdf “http://www.w3.org/1999/02/22-rdf-syntax-ns#”
  xmlns:aistls “http://www.molprof.jp/ontologies/aistlssio.owl#”>
- Subject: RassieOutput, rdf:about: the string as same as specifying in the input RDF
  <aistls:RassieOutput
  rdf:about=”http://www.molprof.jp/ontologies/rassie.rdf#1”>
- Triple for Rassie results:
  Subject: RassieOutput
  Predicate: requiresResultInTextFormat
  Object: string (Rassie results)
  <aistls:requiresResultInTextFormat>Rassie results
  </aistls:requiresResultInTextFormat>

Sample output RDF file is as follows:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:aistls="http://www.molprof.jp/ontologies/aistlssio.owl#">
   <aistls:RassieOutput rdf:about="http://www.molprof.jp/ontologies/rassie.rdf#1">
      <aistls:requiresResultInTextFormat>MODEL 1
      ATOM     1 P     G X   1     12.409 -19.191   9.334
      ATOM     2 O5'   G X   1     11.280 -20.184   8.788
      ATOM     3 C5'   G X   1     11.573 -21.021   7.704
      ATOM     4 C4'   G X   1     10.414 -21.968   7.500
      ATOM     5 C3'   G X   1      9.158 -21.309   7.084
      ATOM     6 O3'   G X   1      8.938 -21.336   5.719
      ATOM     7 O4'   G X   1     10.030 -22.284   8.834
      ……
      ……</aistls:requiresResultInTextFormat>
      <aistls:requiresResultInTextFormat>MODEL 2
      ATOM     1 P     G X   1     12.409 -19.191   9.334
      ATOM     2 O5'   G X   1     11.280 -20.184   8.788
      ATOM     3 C5'   G X   1     11.573 -21.021   7.704
      ATOM     4 C4'   G X   1     10.414 -21.968   7.500
      ATOM     5 C3'   G X   1      9.158 -21.309   7.084
      ATOM     6 O3'   G X   1      8.938 -21.336   5.719
      ATOM     7 O4'   G X   1     10.030 -22.284   8.834
      ……
      ……</aistls:requiresResultInTextFormat>
   </aistls:RassieOutput>
</rdf:RDF>
```

Rassie output RDF

Contact

Please send your queries and comments, if you have, to the address below.
workflow@molprof.jp

Molecular Profiling Research Center for Drug Discovery of AIST plans to listen to user's requests positively, and to make the system better.

Molecular Profiling Research Center for Drug Discovery (MolProf)
Advanced Industrial Science and Technology (AIST)
http://togo.medals.jp
AIST Tokyo Waterfront Bio-IT Research Building
2-4-7 Aomi, Koto-ku, Tokyo, 135-0064, Japan