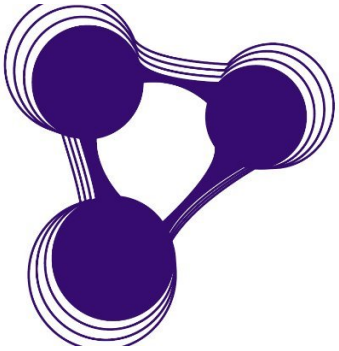

Manual de usuario
AUTOMATON 2.0

Computational & Theoretical Chemistry Group
Universidad Andrés Bello

Descripción

En el manual de usuario de AUTOMATON 2.0 se describe cómo obtener y utilizar el programa, los archivos de entrada necesarios y cómo realizar la ejecución en sistemas operativos con sistemas de gestión de clústeres computacionales y versión local.

Santiago, 27 de enero de 2021

Índice

1. Introducción

1.1. AUTOMATON 2.0

El propósito de AUTOMATON 2.0 es encontrar la estructura más estable de un *clúster* atómico o molecular desde el punto de vista energético, encontrando el mínimo global mediante la exploración estocástica de la Superficie de Energía Potencial (SEP) de estas estructuras mediante optimización, utilizando métodos *ab initio* químico computacionales. Con la fórmula química de un *clúster*, entregada por el usuario, se encuentran los mínimos energéticos locales, el mínimo global y se ordenan estos mínimos de manera decreciente. Esta es una versión actualizada de su antecesor AUTOMATON [4], se utiliza un algoritmo híbrido que combina la teoría de Autómata Celular para la generación de población inicial determinando el conjunto de vecindarios con un algoritmo de dibujo de círculo de punto medio, la representación del autómata se realiza con matrices de 1, 2 y 3 dimensiones y el manejo de éstas es soportado por la librería NumPy [2] de Python, especializada en el cálculo numérico y el análisis de datos. Se combina lo anterior con un algoritmo genético que realiza operaciones de cruzamiento y mutaciones por desplazamiento y permutación de átomos. Esta nueva versión se diseñó y codificó con el paradigma de Programación Orientada a Objetos y se realizó el cambio del lenguaje de programación de Perl a Python, logrando reducir el tiempo de ejecución en un 67 % aproximadamente en comparación con el software original.

1.2. Requisitos

En sistemas operativos Linux/UNIX se recomienda conocimiento básico de comandos bash (*cd, ls, mv, cp...*) del usuario.

AUTOMATON soporta sistemas de gestión de trabajos para clústeres computacionales: SLURM Workload Manager y Sun Grid Engine (SGE); Si se requiere, se podría habilitar el software de programación de trabajos Portable Batch System (PBS) debiendo editar el código fuente en el módulo correspondiente. El programa está escrito en el lenguaje computacional Python en sus versiones 2.7 y 3.5.

Para la optimización geométrica local y cálculos de energía se pueden utilizar los softwares Gaussian 16 [1] u ORCA [3]. El uso del software comercial para química teórica Gaussian está restringido a los usuarios que dispongan de una licencia válida para la versión requerida y será necesaria para cualquiera de los sistemas operativos soportados. Por otro lado, ORCA es de uso gratuito.

2. Instalación

2.1. Instrucciones para instalación en Linux/UNIX

Obtener los archivos desde la rama <py3> de GitHub. Esto se puede realizar con el siguiente comando:

```
git clone --single-branch --branch py3 git@github.com:Rufox/MatrixAutomaton.git
```

Se obtendrán los archivos fuente del programa y se incluye la carpeta `bin` donde se encuentra el archivo binario `Automaton`, el cual permite la ejecución del programa. Además, se incluye el archivo de configuración `Config.in` y el archivo `ENVIAR.sh`, el cual es opcional, debido a que es un sistema por lotes o *modo batch*, es decir, instrucciones de comando que simplifican la ejecución del programa para el usuario. Variará según el tipo de clúster computacional utilizado o si es ejecutado de manera local.

La estructura de los ficheros y archivos obtenidos se representa en la Figura 1.

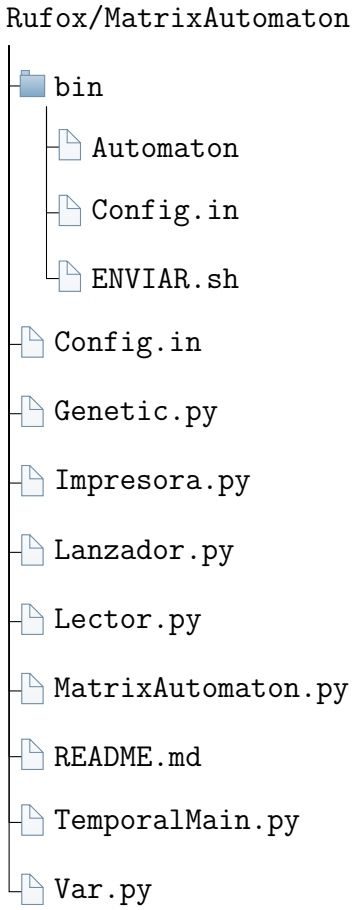

Figura 1: Estructura de ficheros y archivos obtenidos desde Github

El archivo binario `Automaton` se debe ubicar en cualquier carpeta que la arquitectura del sistema computacional permita llamar de manera global al programa

o copiarlo directamente en cada carpeta de trabajo. Se deben otorgar permisos de ejecución al binario y esto se puede realizar con el siguiente comando:

```
chmod 755 InstallationFolderPath/Automaton
```

Donde "InstallationFolderPath" corresponde a la ubicación del archivo.

3. Ejecución

3.1. Instrucciones para uso de AUTOMATON en sistemas operativos Linux/UNIX con sistema de gestión de trabajos

Los sistemas de gestión de trabajos para clústeres computacionales soportados actualmente por AUTOMATON son Slurm Workload Manager y SGE. Se debe detallar cuál se utilizará en el archivo de Configuración llamado Config.in cambiando la variable `job-scheduler`. Esto se detalla en la sección 4.

Se necesitan 3 archivos para que el programa funcione (el binario `Automaton`, `Config.in` y `ENVIAR.sh`) y se recomienda crear una carpeta para cada cálculo que se requiera ejecutar. Por ejemplo, para calcular el mínimo global de C_6H_6, se crea la carpeta C6H6, donde deben incluirse los siguientes archivos:

- `ENVIAR.sh` es el archivo con el que se envía el trabajo al clúster computacional y solo se edita el nombre del trabajo a asignar.
- `Config.in` es el archivo de configuración, desde donde se obtienen las variables a utilizar por AUTOMATON y por Gaussian/Orca cuando sea necesario. La mayoría de las variables de este archivo se mantienen iguales independiente del grupo atómico a definir. Estas variables se definen en la sección 4.

3.2. Instrucciones para uso de AUTOMATON en sistemas operativos Linux/UNIX en modo local

3.2.1. Requerimientos previos para uso en modo local

Es necesario que todas las variables globales de Gaussian estén establecidas en la máquina local. AUTOMATON funcionará llamando a Gaussian por medio del comando a definir en el archivo `Config.in` y será el único archivo de entrada necesario para la ejecución en este modo. Para establecer el modo local, se deben modificar y agregar algunas variables en este archivo:

- `job-scheduler = local`. Variable necesaria, la misma para establecer el uso de slurm o sge.
- `command = g16`. Variable necesaria para modo local, indica el comando a utilizar para llamar al programa Gaussian.
- `parallel = 4`. Variable necesaria para modo local. Indica cuantos cálculos simultáneos se lanzarán en la máquina local.

NOTA = Para la última variable `parallel` se debe tomar en cuenta cuántos procesadores (cores) se utilizarán por cálculo, este numero se multiplicará por el valor de `parallel`. Ejemplo: Si `core = 4` y `parallel = 4`, en total 16 $(4 * 4)$ procesadores serán utilizados simultáneamente.

AUTOMATON no se hace responsable por uso indebido de recursos computacionales. Es decir, NO OCUPE MAS CORES DE LOS PRESENTES EN SU MÁQUINA LOCAL.

Durante la ejecución se crearán archivos llamados `BATCH-0X.slrm` (donde X indica numeración), contienen un compilado de cálculos a lanzar en tándem.

Para visualizar el estado del proceso es necesario revisar la información trasmitida por el archivo `LOGS`. De manera opcional, el usuario puede utilizar los comandos `top` o `htop` para ver el estado de los cálculos.

4. Archivos de entrada

Config.in es el archivo de configuración, desde donde se obtienen las variables a utilizar por AUTOMATON y por Gaussian/ORCA cuando sea necesario. La mayoría de las variables de este archivo se mantienen iguales independiente del grupo atómico a definir, las variables que deben cambiarse en el archivo de entrada son:

- Número de estructuras ($5N$, N = cantidad de átomos). Por ejemplo, si $N = 12$: `numb_conf = 60`
- `job-scheduler = slurm`. Sistema de gestión de trabajos (slurm, sge, local).
- `chemical_formula = B 8 Be 4`. Fórmula química para el sistema.
- `software = gaussian`. Software que será utilizado (gaussian u orca).

Configurando el programa para paquetes químicos:

- `core = 4`
  `memory = 4`.
  Procesador y memoria (GB) que será utilizado para cada cálculo:
- `charge_multi = 0 1`. Carga y multiplicidad del candidato (separados por un espacio).
- `header = PBE1PBE/SDDAll scf=(maxcycle=512) opt=(cartesian,maxcycle=512)`. Palabras claves de trabajo gaussian. Por ejemplo: nivel de teoria, opt (optimizacion), etc.

El archivo de entrada opcional `ENVIAR.sh` se detalla en la sección 6.

4.1. Variables Ocultas

El programa incluye algunas variables ocultas que permiten la modificación de parámetros definidos por defecto. De ser necesaria la modificación de alguna de estas variables, se puede incluir en el archivo de configuración en la sección [GENERAL]. Estas variables ocultas se detallan a continuación:

- `Pcent1D = 0.1`. Numero de individuos por generación a crear con autómata que cumplan ser de 1 dimensión. Valor entre [0,1].
  Valor asignado por default: 10 %.

- `Pcent2D = 0.3`. Numero de individuos por generación a crear con autómata que cumplan ser de 2 dimensiones. Valor entre [0,1].
  Valor asignado por default: 30 %.

NOTA = El porcentaje de individuos en formato 3D a crear se ajustará dependiendo de los valores de `Pcent1D` y `Pcent2D`.

- `PcentToCreate = 0.1`. Porcentaje de la población formada exclusivamente por autómata en cada generación del algoritmo genético. Valor entre [0,1].
  Valor asignado por default: 10 %.

- `PcentToMutate = 0.2`. Porcentaje de la población formada exclusivamente por operaciones de mutación en cada generación del algoritmo genético. Valor entre [0,1].
  Valor asignado por default: 20 %.

NOTA = El porcentaje de individuos producto de operaciones de recombinación se obtiene automáticamente gracias a los valores de `PcentToMutate` y `PcentToCreate`.

- `PcentAtomosMutadosMovimiento = 0.3`. Porcentaje de átomos que sufrirán un movimiento aleatorio en los procesos de mutación. Valor entre [0,1].
  Valor asignado por default: 30 %.

- `alphaNumber = 3`. Valor de potencia `alphaNumber` en la formula $exp(-alphaNumber * prob)$, donde:

  $$prob = \frac{(E_i - E_{\text{mín}})}{(E_{\text{máx}} - E_{\text{mín}})}$$

  donde:

  - $E_{\text{mín}}$: energía mínima
  - $E_{\text{máx}}$: energía máxima

- E_i: energía de cada sistema

Valores altos aumenta el elitismo a la hora de selección de buenas estructuras por ciclo.
Valor asignado por default: 3.

- `PcentBestFitness = 0.5`. Porcentaje de corte para seleccionar estructuras de elite por ciclo. Se le aplica a la fórmula anterior. Valor entre [0-1].
  Valor asignado por default: 50 %.

- `PCentCloseness = 1`. Método de control para obligar a unir estructuras creadas por medio de autómata. Entiéndase como un porcentaje de cercanía de los átomos, si es mayor los átomos se alejarán, menor y se acercarán. Valor [0,inf[.
  Valor asignado por default: 1.

- `maxConvergencia = 9`. Numero de ciclos necesarios para finalizar búsqueda del clúster/molécula. El programa terminara correctamente solo cuando una estructura se mantenga como mejor (menor energía) durante un numero de ciclos. Valor [1-inf[.
  Valor asignado por default: 9.

- `shuffleElements = 0`. Permite agregar átomos de manera no aleatoria. El orden de agregación se define en el archivo Config.in al ingresar al formula química, se leerá de derecha a izquierda. Ejemplo: H 2 O 3 C 1 agregará primero 1 C, luego 3 O y finalmente 2 H. Valor asignado por default: 1.

- En caso de que el programa deje de funcionar por algún motivo, se puede agregar la variable `reset = 1` al archivo Config.in en la sección [GENERAL]. Luego de agregarla y guardar los cambios, volver a lanzar el programa con el archivo ENVIAR.sh.
  `reset = 1`

Nota general: Respete los espacios de separación entre el símbolo =.

- La manera correcta: software = gaussian
- La manera equivocada: software=Gaussian

5. Archivos de salida

Luego de una ejecución exitosa del programa, se generarán varios archivos de salida en su directorio de trabajo:

- `01Finalscoords.xyz`: Archivo de formato XYZ que entrega las coordenadas finales de cada especie ordenado de mayor a menor energía.

- `Original*D_i_j`: Población inicial creada con el programa de matrices.

    - * = cantidad de dimensiones (1, 2 o 3).
    - i = ciclo.
    - j = enumeración de conformación.
- `Child_i_j`: Se generan con la recombinación del sistema. Se crean a partir de los archivos "`Original*D_i_j`".
- Mutación:
    - `MutD_i_j`: Mutación de desplazamiento.
    - `MutI_i_j`: Mutación de tipo intercambio.
- Los archivos Original, Child y MutD/MutI tienen 3 extensiones diferentes:
    - `archivo.com`: Archivo de entrada de gaussian, coordenadas.
    - `archivo.slrm`: Archivo de entrada del cluster.
    - `archivo.log`: Archivo de salida de gaussian.
- `Precoords_i.xyz`: Archivo de formato xyz con las estructuras no optimizadas por ciclo.
- `PostCoords_i.xyz`: Archivo de formato xyz con las estructuras optimizadas localmente por ciclo.
- `LOGS`: Archivo de registro de la ejecución del programa. Aquí se registra la fecha y hora de inicio y término de ejecución del programa, los ciclos y estado de convergencia y la información sobre las energías mínimas encontradas y el archivo donde se encontró.

6. Ejemplos

6.1. Ejecución en SLURM Workload Manager

El siguiente ejemplo muestra la configuración que debe ser realizada para calcular el mínimo global del H_2O utilizando el sistema de gestión de trabajos SLURM Workload Manager y el software Gaussian 16 para las optimizaciones locales de energía. El contenido del archivo de entrada llamado Config.in se detalla en la Figura 2. Configura las variables de entrada del programa, por ejemplo, la fórmula química del *clúster* atómico/molecular a explorar, por consecuencia también la cantidad N de átomos del *clúster* y la cantidad de conformaciones creadas por ciclo.

```
[GENERAL]
##numb_conf = 5N, N = 2 + 1
numb_conf = 15

job-scheduler = slurm

[CLUSTER]

chemical_formula = H 2 O 1

[SOFTWARE]
software = gaussian

core = 4
memory = 4

charge_multi = 0 1

header = PBE1PBE/SDDAll scf=(maxcycle=512) opt=(cartesian,maxcycle=
512)integral=NoXCTest
```

Figura 2: Archivo de entrada Config.in para SLURM Workload Manager

Se incluye el archivo ENVIAR.sh (Figura 3) que permite configurar algunos parámetros del clúster computacional y enviarlo de una manera simple. Aquí el usuario sólo debe editar la variable `--job-name` para identificar el trabajo enviado en la cola de trabajos en ejecución.

```
#!/bin/bash
#SBATCH --job-name=MA_H2O
#SBATCH --partition=general
#SBATCH --nodes=1
#SBATCH -c 1
#SBATCH --output=trash

srun ./Automaton Config.in general
```

Figura 3: Archivo de entrada ENVIAR.sh para SLURM Workload Manager

Ubicados en la carpeta de trabajo, se ejecuta el programa con el siguiente comando:

```
sbatch ENVIAR.sh
```

Para ver los procesos en ejecución generados por el programa se utiliza el comando

`squeue`. Luego de una correcta ejecución, se crean algunos archivos de salida detallados anteriormente. El archivo de salida donde se visualizan las estructuras mínimas locales y la estructura mínima global con sus respectivas energías y coordenadas, se llama `01FinalsCoords.xyz` (Figura 4) donde el mínimo global va a ser la última estructura registrada y para el ejemplo de la molécula de agua se obtuvo lo siguiente:

```
3
Original2D_0_2         E = -17.1991156012 H
O         0.0         0.0         0.1095
H         -0.0         0.8007         -0.4378
H         -0.0         -0.8007         -0.4378
3
Child_3_6         E = -17.1991156013 H
H         0.8007         -0.4378         0.0
O         0.0         0.1094         -0.0
H         -0.8007         -0.4378         -0.0
```

Figura 4: Archivo de salida 01FinalsCoords.xyz para H2O utilizando SLURM

En este ejemplo se utiliza el software gráfico para la visualización de cálculos de química cuántica Chemcraft para visualizar la estructura mínima global encontrada para el H2O (Figura 5).

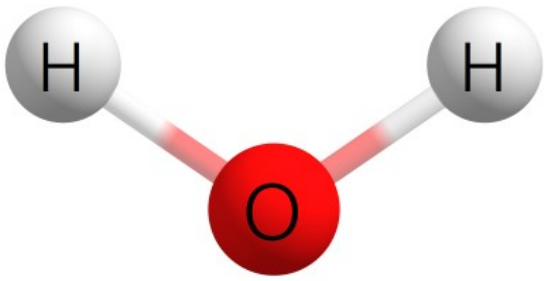

Figura 5: Mínimo global H2O

El archivo de registro `LOGS`, muestra la fecha y hora de inicio y término de ejecución del programa, además del proceso por ciclos, el estado de convergencia, los mínimos encontrados y en qué archivo se encontraron. Para el caso del ejemplo, este archivo de registro se detalla en la Figura 6. El mínimo global fue encontrado en el archivo `Child_3_6`, es decir, se encontró luego de haber realizado una operación genética de recombinación, específicamente la conformación n° 6 del ciclo 3. con una energía de $-17,1991156013H$.

```
Iniciado el programa @ 08/14/2020 17:17:00

Ciclo = 0
Nuevo Minimo! -> Original2D_0_2
Energia :-17.1991156012 H
Convergencia 1 de 9

Ciclo = 1
Convergencia 2 de 9

Ciclo = 2
Convergencia 3 de 9

Ciclo = 3
Nuevo Minimo! -> Child_3_6
Energia :-17.1991156013 H
Convergencia 1 de 9

Ciclo = 4
Convergencia 2 de 9

Ciclo = 5
Convergencia 3 de 9

Ciclo = 6
Convergencia 4 de 9

Ciclo = 7
Convergencia 5 de 9

Ciclo = 8
Convergencia 6 de 9

Ciclo = 9
Convergencia 7 de 9

Ciclo = 10
Convergencia 8 de 9

Ciclo = 11
Convergencia 9 de 9

Proceso Finalizado @ 08/14/2020 17:17:00
```

Figura 6: Archivo de salida LOGS para H2O utilizando SLURM

6.2. Ejecución en Sun Grid Engine

Si se quiere realizar el mismo cálculo en un cluster con el sistema SGE, se modifica la variable `job-scheduler` en `Config.in`:

```
[GENERAL]
##numb_conf = 5N, N = 2 + 1
numb_conf = 15

job-scheduler = sge

[CLUSTER]

chemical_formula = H 2 O 1

[SOFTWARE]
software = gaussian

core = 4
memory = 4

charge_multi = 0 1

header = PBE1PBE/SDDAll scf=(maxcycle=512) opt=(cartesian,maxcycle=
512)integral=NoXCTest
```

Figura 7: Archivo de entrada Config.in para Sun Grid Engine

Y el archivo `ENVIAR.sh` se modifica con la homologación de los comandos de SLURM:

```
#! /bin/bash
#
#$ -cwd
#$ -j y
#$ -N MA_H2O
#$ -o trash
#$ -S /bin/bash
#$ -pe solouno 1
#

./Automaton Config.in QUEUE
```

Figura 8: Archivo de entrada ENVIAR.sh para Sun Grid Engine

Ubicados en la carpeta de trabajo, se ejecuta el siguiente comando:

```
qsub -q QUEUE_1 ENVIAR.sh
```

Donde `QUEUE\_1` debe tener permisos de envío. Para ver los procesos generados por el programa se utiliza el comando `qstat`.

6.3. Ejecución en modo local

Para realizar una ejecución en modo local se necesita el archivo binario `Automaton` y el archivo de configuración `Config.in` como se detalló en la sección 3.2.1. El ejemplo del archivo de configuración se representa en la Figura 9.

```
[GENERAL]
##numb_conf = 5N, N = 2 + 1
numb_conf = 15

job-scheduler = local

command = g16

[CLUSTER]

chemical_formula = H 2 O 1

[SOFTWARE]
software = gaussian

core = 4
memory = 4
parallel = 4

charge_multi = 0 1

header = PBE1PBE/SDDAll scf=(maxcycle=512) opt=(cartesian,maxcycle=
512)integral=NoXCTest
```

Figura 9: Archivo de entrada Config.in para modo local

Para realizar la ejecución se utiliza el comando:

```
./Automaton Config.in
```

Alternativamente, el usuario puede ejecutar AUTOMATON en segundo plano utilizando uno de los siguientes métodos:

```
nohup ./Automaton Config.in > out.log
```

```
setsid ./Automaton Config.in > out.log
```

Referencias

[1] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox. Gaussian˜16 Revision C.01, 2016. Gaussian Inc. Wallingford CT.

[2] Charles R. Harris, K. Jarrod Millman, St'efan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fern'andez del R'ıo, Mark Wiebe, Pearu Peterson, Pierre G'erard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[3] Frank Neese. Software update: the orca program system, version 4.0. *WIREs Computational Molecular Science*, 8(1):e1327, 2018. doi: https://doi.org/10.1002/wcms.1327. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wcms.1327.

[4] Osvaldo Yañez, Rodrigo Báez-Grez, Walter A. Rabanal-León Diego Inostroza, Jorge Garza Ricardo Pino-Rios, and William Tiznado. AUTOMATON: A Program That Combines a Probabilistic Cellular Automata and a Genetic Algorithm for Global Minimum Search of Clusters and Molecules. *J. Chem. Theory Comput.*, 15(2):1463—1475, 2019.