Stoichiometry Determination for Mass-spectrometry Data: the Interval Case

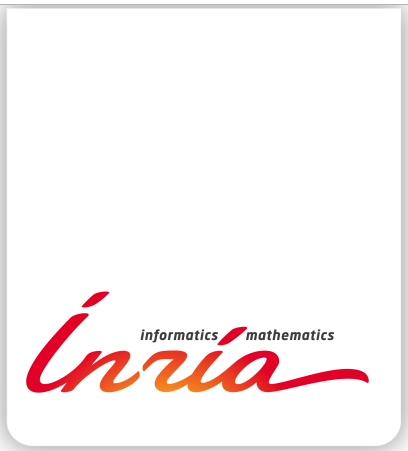
Deepesh Agarwal, Frédéric Cazals, Noël Malod-Dognin

Stoichiometry Determination for Mass-spectrometry Data: the Interval Case

Deepesh Agarwal, Frédéric Cazals and Noël Malod-Dognin

RESEARCH REPORT
N° 8101
October 2012
Project-Team ABS

ISSN 0249-6399 ISRN INRIA/RR--8101--FR+ENG



Stoichiometry Determination for Mass-spectrometry Data: the Interval Case

Deepesh Agarwal, Frédéric Cazals and Noël Malod-Dognin

Project-Team ABS

Research Report n° 8101 — version 2 — initial version October 2012 — revised version February 2013 — 49 pages

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Abstract: In structural proteomics, given the individual masses of a set of protein types and the exact mass of a protein complex, the *exact stoichiometry determination problem (SD)*, also known as the money-change problem, consists of enumerating all the stoichiometries of these types which allow to recover the target mass. If the target mass suffers from experimental uncertainties, the *interval SD problem* consists of finding all the stoichiometry vectors compatible with a target mass within an interval.

We make contributions in two directions. From an algorithmic standpoint, we make two modifications to exact SD algorithms, to inherently address the interval SD problem. The first modification concerns the classical tree-like exploration, resulting in algorithm `DIOPHANTINE`, which solves the interval SD problem using constant-memory space. The second modification concerns the classical dynamic programming based algorithm, resulting in algorithm `DP++`, which solves the interval SD problem in an output sensitive manner. From an applied perspective, we raise three points. First, we show that `DIOPHANTINE` and `DP++` yield an improvement from 3 to 4 orders of magnitude over state-of-the-art exact SD algorithms, for typical protein complexes facing uncertainties on the target mass in the range 0.1-1%. It is shown that this gain comes from the ability of `DP++` and `DIOPHANTINE` to avoid redundant calculations occurring when solving the interval SD problem as a sequence of exact SD problems. Second, we show that `DIOPHANTINE` behaves like an output-sensitive algorithm—especially when the interval width increases, albeit such a property cannot be expected in general. Third, from a biological perspective, using a panel of biological complexes (eukaryotic translation factor, yeast exosome, 19S proteasome sub-unit, nuclear pore complex), we stress the importance of enumeration, even at a null noise level. We also propose a representation of all solutions of an interval SD problem using a directed acyclic graph stressing the prominent protein types.
The programs accompanying this paper are available from http://team.inria.fr/abs/addict/.

Key-words: Protein complex, mass spectrometry, denumerant, dynamic programming, linear diophantine equation, enumeration, output sensitive algorithm.

Sur la détermination de la stoichiométrie en spectrométrie de masse: détermination pour un intervalle de masses

Résumé : En protéomique structurale, étant données les masses exactes d'un ensemble de protéines et la masse exacte d'un complexe, le problème de la *détermination de la stoechiométrie* (SD), consiste à énumérer les stoechiométries de ces protéines permettant de reconstituer la masse du complexe. Si celle-ci présente une incertitude, il s'agit d'énumérer les stoechiométries compatibles avec une masse appartenant à un intervalle. Nous présentons des contributions dans deux registres.

D'un point de vue théorique, nous apportons des modifications aux algorithmes exacts de SD, afin de résoudre le problème par intervalle. La première modification concerne l'algorithme de recherche exhaustif, et conduit à l'algorithme `DIOPHANTINE` qui travaille à mémoire constante. La seconde concerne l'algorithme état de l'art utilisant la programmation dynamique, et conduit à l'algorithme `DP++`, lequel est sensible à la sortie. D'un point de vue appliqué, nous présentons trois résultats. Tout d'abord, nous montrons que `DIOPHANTINE` et `DP++` sont plus rapides de 3 à 4 ordres de grandeur que l'algorithme état de l'art résolvant le problème exact, pour des complexes protéiques typiques présentant des incertitudes de l'ordre de 0.1% a 1%. Nous montrons que ce gain tient à l'évitement de calculs redondants lorsque l'algorithme exact est appellé sur chacune des masses d'un intervalle. D'autre part, nous montrons que `DIOPHANTINE` a un comportement d'autant plus *sensible à la sortie* que la taille de l'intervalle augmente. Enfin et d'un point de vue biologique, en étudiant plusieurs complexes protéiques (facteur de traduction eucaryote, *exosome* de la levure, sous-unité 19S du protéasome, pore nucléaire), nous soulignons l'importance des solutions multiples, même à un niveau d'incertitude nul. Nous introduisons également une représentation de toutes les solutions d'un problème par intervalle, mettant en evidence les types de protéines les plus utilisés.

Mots-clés : Complexes protéiques, spectrométrie de masse, dénumérant, programmation dynamique, équation diophantienne, énumération, algorithmes sensibles à la sortie.

1 Introduction

1.1 Structural Proteomics, Integer Partitions and Knapsack Problems

Mass spectrometry. Biology rests on multimeric assemblies, so that reconstructing models of such assemblies is a central problem in (computational) structural biology. While the structure determination of proteins and small complexes has increased at a fast pace thanks to structural proteomics projects [Jan07], that of larger complexes, which typically do not crystallize or are too big for nuclear magnetic resonance studies has remained elusive. Fortunately, mass spectrometry (MS), a technique initially developed in chemistry to infer the mass-to-charge ratio of molecules, is in the process of filling this gap, allowing to study complexes that vary in mass, size, solubility, and bound/unbound states [SR07, BR11]. In a nutshell, MS first requires isolating sample complexes from the cellular environment, a process typically resorting to Tandem Affinity Purification [ea01]. Operating on these samples in solution, a soft ionization technique such as electrospray ionization or matrix-assisted laser desorption/ionization allows ionizing the proteins or protein assemblies, while preserving non covalent interactions—a key feature to study proteins and complexes [SR07]. The ions produced are then passed to a mass selection instrument (time of flight analyzer, ions traps, orbitraps, ...), recording information based on the ration mass-to-charge of the ions.

In classical *time-of-flight* MS, these primary ions are sorted by a mass analyzer according to their mass over charge (m/z) ratio, and the relative abundance of the samples are measured by a detector. Unfortunately, for a solution containing a mixture of complexes, different complexes with identical m/z ratio may contribute to the same peak in the mass spectrum. To disentangle such cases, tandem mass spectrometry (tandem MS) can be used. In this refined protocol, the precursor ions undergo a collision induced dissociation, which consists of removing highly charged peripheral sub-units thanks to gas collisions, so that the product ions can indeed be sorted by m/z ratio.

MS can also be used to gain information on the sub-complexes of an assembly [THS+08]. Denaturating solvent or high salt concentrations can be used to break the complex into pieces, which are then identified by MS or tandem MS. In extreme conditions, complete denaturation occurs, so that the masses of the individual components can be determined. In milder conditions, multiple overlapping complexes are generated, providing information on the building blocks.

Stoichiometry determination. Upon determining the masses (those of the complexes in a mixture, or those of the sub-complexes of an assembly), the stoichiometry determination (SD) problem consists of determining the stoichiometry of the protein types accounting for each target mass. Denoting p the number of protein types, w_i and s_i the mass and the stoichiometry of the ith protein type respectively, one wishes to reconstruct the target mass M as follows:

$$\sum_{i=1,\ldots,p} s_i \; w_i = M. \tag{1}$$

This problem, which was first addressed in [DF80] using an algorithm exploring all possible stoichiometry vectors in a tree-like fashion, is also known as the *money exchange* problem, for obvious reasons [BL05b]. Mathematically and since the masses are integers (expressed say in Daltons [1]), SD is related to the theory of integer partitions [Com74], to linear diophantine equations [Sma98], and is also coupled to so-called knapsack and subset sum problems [Pis05].

[1] The unified atomic mass unit, also known as Dalton, is approximately equal to the mass of one neutron or one proton.

Uncertainties in the mass determination. Practically, two difficulties are faced in mass spectrometry. First, because the proteins involved in different copies of a complex may have undergone different post-translational modifications [2] [Kel04], there might be a discrepancy between the masses of the sub-units. Second, any mass experimentally determined is generally larger than the theoretical one, due to extra molecules (solvent and electro-spray buffer molecules) sticking to the molecule or complex analyzed. The magnitude of this second source of error actually depends on the type of ionization technique used and on the type of sample.

Mass analyzers such as orbitraps, which were initially designed to handle small molecules, typically achieve relative mass error of 0.005% (50 part-per-million or ppm). Interestingly, it has been reported recently that upon modifying the instruments, in particular the ability to completely desolvate the samples [RDD+12], intact macromolecular assemblies with molecular weights of the order 1 MDa were amenable to studies with the same precision. Yet, this tour de force has been achieved so far on very few macro-molecular complexes as follows, IgG antibody (146 kDa), bacteriophage HK97 (253 kDa), yeast 20S proteasome (730 kDa), *E. coli* GroEL (801 kDa).

On the other hand, classical Time of Flight (TOF) analyzers accommodate a much wider range of samples, and continue to be used for characterization of large macromolecular assemblies [ZCGB13]. Using such instruments, a relative mass error as high as $\sim 1\%$ are typical for hetero-oligomeric complexes of up to 1 MDa [THS+08, MR12].

The interval problem. To account for the uncertainties on the masses just discussed, let ε be a positive integer corresponding to the mass discrepancy to be accommodated. For the sake of conciseness, the relative mass error r is called the *noise level* in the sequel, the number ε being defined by $\varepsilon = r \times M$. Instead of solving Eq. (1), we are instead interested in the following enumeration problem

$$\text{Report all stoichiometry vectors } S = (s_1, \ldots, s_p) \text{ such that } \mid \sum_{i=1,\ldots,p} s_i w_i - M \mid \leq \varepsilon. \quad (2)$$

For the sake of conciseness, the previous problem uses symmetric errors terms $\pm\varepsilon$ with respect to the target mass M. However, an interval $[M-\varepsilon_1, M+\varepsilon_2]$ with $\varepsilon_1 \neq \varepsilon_2$ can be accommodated at the cost of trivial modifications of our algorithms.

1.2 Contributions

To the best of our knowledge, the question of mass accuracy for the complex composition to be determined has only been touched upon indirectly. In [BLM+08], the interval SD problem is solved by repeatedly calling the exact SD algorithm on each target mass in the interval. In [THS+08], 100 hypothetical complexes from 6 to 14 sub-units have been created with masses in the range 10-50kD, and 100 dimers, trimers or tetramers were generated from these complexes. With an error rate of 1%, it is observed that no tetramer has a unique stoichiometry vector.

In this context, we make two contributions. First, we present a constant memory space algorithm (`DIOPHANTINE`), and an output sensitive dynamic programming based algorithm (`DP++`), both inherently addressing the interval SD problem. For `DIOPHANTINE`, we also show that sorting the masses w_i yields a tree of optimal size, an observation which had not been made previously. Second, we present a detailed experimental study on various biological and synthetic datasets, which results in three conclusions. The first observation is that for biological datasets, enumeration does matter even for a moderate (and sometimes even null) noise level. The second

[2] The chemical modification of a protein after its bio-synthesis by the ribosome.

finding is that our algorithms `DIOPHANTINE` and `DP++` outperform state-of-the art dynamic programming based approaches by three to four orders of magnitude, for a noise level in the range 0.1% to 1%, which is typically faced in structural proteomics. Not surprisingly, we show that this improvement owes to the ability of our algorithms to avoid redundant calculations which are carried out when calling the exact dynamic programming based algorithm on each target mass in an interval. The last one is that `DIOPHANTINE` actually exhibits an output sensitive behavior. Thus and interestingly, we show that one of the very first algorithms designed to solve the exact SD problem [DF80] can be modified not only to solve the interval SD problem, but also to outperform advanced strategies based on dynamic programming.

2 Theory and Algorithms

In section 2.1, we recall the rich background of knapsack and integer partition problems. In section 2.2, we present classical counting results, and derive bound on the number of solutions for the SD problem.

2.1 Denumerants, Unbounded Knapsack and Subset-sum Problems

In the following, we consider a vector $\mathbf{W} = \{w_1, \ldots, w_p\}$, of positive integers or real numbers. A stoichiometry vector is denoted $\mathbf{S} = \{s_1, \ldots, s_p\}$. The vector is called positive if $s_i > 0$ for all i, and non-negative if $s_i \geq 0$ for all i.

Denumerants. Assume that the vector $\mathbf{W}$ contains integers. In combinatorics, the number of non-negative integer solutions to Eq. (2) is known as the *denumerant* $D(M)$ of the target mass. It has been known since Bell [Bel43] that if $\text{lcm}(\mathbf{W})$ denotes the least common multiple of the w_is, then, for each $b \in \{0, \ldots, \text{lcm}(\mathbf{W}) - 1\}$, if $M = m\ \text{lcm}(\mathbf{W}) + b$, the denumerant is a polynomial in m of degree $p - 1$, that is:

$$D(m\ \text{lcm}(\mathbf{W}) + b) = \sum_{i=0,\ldots,k-1} c_i m^i. \tag{3}$$

More generally, the number of solutions reads from the power series expansion of the generating function $1/\Pi_i(1 - X^{w_i})$, but the difficulty precisely relies in extracting the coefficient of X^n in that expansion. In analytic combinatorics [FS09, Chapter IV], a classical result obtained by singularity analysis states that if $\gcd(\mathbf{W}) = 1$, one asymptotically gets

$$D(M) \sim \frac{M^{p-1}}{(p-1)! w_1 \ldots w_p}. \tag{4}$$

Upper and lower bounds on the denumerant have also been obtained based on binomial coefficients solution of certain recurrence relations [Agn02]. However, these bounds are not of real interest for the SD problem for two reasons: first, the bounds concern a particular denumerant rather than the solutions corresponding to an interval, as specified by Eq. (2); second, as we shall see with experiments, the bounds returned are not tight.

Frobenius numbers. The asymptotic behavior given by Eq. (4) apparently contradicts the existence of (small) masses which cannot be decomposed, this issue being related to the so-called Frobenius number and its generalizations. The Frobenius number $g_0(\mathbf{W})$ is defined as the largest integer which cannot be represented as a non-negative integer combination of the integers in $\mathbf{W}$

[Alf05]. While explicit formula are known up to $M = 3$, only upper and lower bounds are known in general [Alf05]. Computing the Frobenius number is a NP-hard problem [RA96] for which pseudo polynomial time algorithms exist, such as the one by Round-Robin [BL05b]. (The running time of such an algorithm is polynomial with respect to the numerical values of the input data, but exponential in their bit-length.)

A related number is the positive Frobenius number $g_0^+(\mathbf{W})$, namely namely the largest integer which does not admit any positive integer solution. As observed in [BDF$^+$10, Lemma 4], both numbers satisfy:

$$\text{Either } g_0^+(\mathbf{W}) = g_0(\mathbf{W}) = 0, \text{ or } g_0^+(\mathbf{W}) = g_0(\mathbf{W}) + \sum_{i=1,\ldots,p} w_i. \tag{5}$$

Unbounded knapsack (UKP) and subset-sum (SSP). Assume that the weights in $\mathbf{W}$ are real numbers coding the weights of p object types, and that each object also has a value $v_i \in \mathbb{R}^+$. Given a target mass M, the unbounded knapsack problem (UKP) consists of finding for each type the integral quantity $s_i \geq 0$, such that the corresponding sum of values is maximum while the corresponding sum of weights does not exceed M. This is formally defined by the following integer programming model:

$$\text{Maximize } \sum_{i=1}^{p} s_i \; v_i, \tag{6}$$

$$\text{Under the constraint } \sum_{i=1}^{p} s_i \; w_i \leq M. \tag{7}$$

The special case where $w_i = v_i$ holds is known as the subset-sum problem (SSP), and consists in finding the quantities s_i such that the corresponding sum of weights is the closest to M, without exceeding it. Changing Eq. (7) into an equality allows subset-sum to solve the exact SD problem of Eq.(1).

UKP and SSP: algorithms and complexity. The optimization version of SSP is one of the first problem proved to be NP-Hard [Kar72], but it is also known to be pseudo-polynomial. An example of such pseudo-polynomial time algorithms is well known Bellman recursion [Bel57], which solves UKP in O(M p) time. More recently, an output sensitive algorithm solving SSP has been developed [BL05a]. Denoting w_1 the smallest mass, the algorithm has complexity $O(p\ w_1\ D(M))$, and relies on a data structure of size $O(p\ w_1)$. We shall use our implementation of this algorithm, called DECOMP, as main contender.

Other approaches have been developed to solve UKP, either based on dynamic programming, on branch and bound, or on a combination of both [PYA09]. Interested readers are referred to [PKP04] for detailed review on the different knapsack problems and on the different techniques used to solve them.

2.2 UKP and SSP: on the Number of Solutions

Non-negative solutions. The number of solutions #UKP$(p, \mathbf{W}, M)$ to an unbounded knapsack problem or of a subset-sum problem that is constrained by Eq. (7) was first studied by Beged-Dov [Bd72], and the bounds were later refined by Padberg [Pad71] and Lambe [Lam74].

Denote $\#SD(p, \mathbf{W}, M, \varepsilon)$ the number of solutions to the SD problem defined by the vector $\mathbf{W}$, and by the target mass $M \pm \varepsilon$. The knowledge of this value is of interest in the context of

stoichiometry determination, to avoid generating a large number of solutions — which would be impossible to analyze. From the exact number of solutions to an UKP problem, one gets

$$\#SD(p, \mathbf{W}, M, \varepsilon) = \#\text{UKP}(p, \mathbf{W}, M + \varepsilon) - \#\text{UKP}(p, \mathbf{W}, M - \varepsilon - 1). \tag{8}$$

However, the previous works [Bd72, Pad71, Lam74] only provide bounds. If # counts an exact number of solutions, denote $\underline{\#}$ and $\overline{\#}$ a lower and an upper bound on #, respectively. Using upper and lower bounds on $\#\text{UKP}(p, \mathbf{W}, M)$, one gets the following lower bound

$$\#SD(p, \mathbf{W}, M, \varepsilon) \geq \underline{\#}\text{UKP}(p, \mathbf{W}, M + \varepsilon) - \overline{\#}\text{UKP}(p, \mathbf{W}, M - \varepsilon - 1), \tag{9}$$

together with the following upper bound

$$\#SD(p, \mathbf{W}, M, \varepsilon) \leq \overline{\#}\text{UKP}(p, \mathbf{W}, M + \varepsilon) - \underline{\#}\text{UKP}(p, \mathbf{W}, M - \varepsilon - 1). \tag{10}$$

In a nearby vein, a fully polynomial-time approximation scheme (FPTAS) has been derived to estimate the number of solutions of any knapsack problem [GKM$^+$]. It provides an upper bound that is at most $(1 + \delta)\#$, where δ is an input parameter, and has a time complexity that is polynomial in M and in $1/\delta$. Using the properties of this upper bound, we can easily derive a lower bound and use them with Eqs. (9) and (10) to bounds SD. In the experiments section, we assess the ability of these strategies to estimate the number of solutions of real stoichiometry determination problems.

Positive solutions. In line with the definition of the shifted Frobenius number of Eq. (5), the question of estimating the number of positive solutions is also of interest, in particular for the biological cases where positive stoichiometries are prescribed for selected protein types. To estimate the number of such solutions, it is sufficient to compute the previous bounds with a target mass reduced by the minimal imposed stoichiometries.

3 The Interval Stoichiometry Determination Problem

3.1 Tree Like Enumeration

The exact case. A pedestrian way to compute the denumerant consists of exhaustively trying all solutions [DF80, MHFH10, MR12]. To solve the interval SD problem of Eq. (2) using the same principle, assume that the stoichiometry vectors are built incrementally—say from left to right, and that the stoichiometry vector $\mathbf{S}$ has been computed up to index i (see also Fig. 1). Under this assumption, we define the remaining mass to be accounted for by the $p - i$ proteins whose stoichiometry has to be determined by

$$M_0^-[\mathbf{S}] = M, \text{ and } M_i^-[\mathbf{S}] = M - \sum_{j=1,\ldots,i} s_j w_j \text{ for } i = 1, \ldots, p-1, \; M_i^-[\mathbf{S}] \geq 0. \tag{11}$$

Then, denoting $a \mid b$ the fact that the integer a divides the integer b, the naive counting strategy for the denumerant consists of computing the following nested sum:

$$\sum_{s_1=0}^{\lfloor M/w_1 \rfloor} \sum_{s_2=0}^{\lfloor M_1^-[\mathbf{S}]/w_2 \rfloor} \cdots \sum_{s_{p-1}=0}^{\lfloor M_{p-2}^-[\mathbf{S}]/w_{p-1} \rfloor} I(w_p : s_1, s_2, s_3 \ldots . s_{p-1}) \tag{12}$$

with

$$I(w_p : s_1, s_2, s_3 \ldots . s_{p-1}) = \begin{cases} 1 & w_p | M_{p-1}^-[\mathbf{S}] \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

The interval case. To solve Eq. (2) using the generalized sum of Eq. (12), observe that upon reaching the last sigma, the following condition holds:

$$M - \varepsilon - \sum_{j=1,\ldots,p-1} s_j w_j \leq s_p w_p \leq M + \varepsilon - \sum_{j=1,\ldots,p-1} s_j w_j, \tag{14}$$

or equivalently

$$\lceil \frac{M - \varepsilon - \sum_{j=1,\ldots,p-1} s_j w_j}{w_p} \rceil \leq s_p \leq \lfloor \frac{M + \varepsilon - \sum_{j=1,\ldots,p-1} s_j w_j}{w_p} \rfloor \tag{15}$$

Therefore, denoting $\#(\cdot)$ the length of an integer interval, counting the number of stoichiometry vectors solving the SD problem is done by

$$\sum_{s_1=0}^{\lfloor (M+\varepsilon)/w_1 \rfloor} \sum_{s_2=0}^{\lfloor (M+\varepsilon)_1^-[\mathbf{S}]/w_2 \rfloor} \cdots \sum_{s_{p-1}=0}^{\lfloor (M+\varepsilon)_{p-2}^-[\mathbf{S}]/w_{p-1} \rfloor} I'(w_p : s_1, s_2, s_3 .... s_{p-1}) \tag{16}$$

with

$$I'(w_p : s_1, s_2, s_3 .... s_{p-1}) = \#_{s_p}(\lceil (M-\varepsilon)_{p-1}^-[\mathbf{S}]/w_p \rceil \leq s_p \leq \lfloor (M+\varepsilon)_{p-1}^-[\mathbf{S}]/w_p \rfloor) \tag{17}$$

Note that when $\varepsilon = 0$, the equations (16-17) and (12-13) coincide.

Remark 1. *For the sake of conciseness, given a stoichiometry vector* $\mathbf{S}$*, let* $\sum_{\mathbf{S}} = \sum_{s_i \in \mathbf{S}} s_i w_i$*. Also, let the* remaining mass *associated to* $\mathbf{S}$ *be defined as* $m = M + \varepsilon - \sum_{\mathbf{S}}$*. One has:*

$$M - \varepsilon \leq \sum_{\mathbf{S}} \leq M + \varepsilon \Leftrightarrow 0 \leq \sum_{\mathbf{S}} -M + \varepsilon \leq 2\varepsilon \Leftrightarrow 0 \leq m \leq 2\varepsilon. \tag{18}$$

These equivalent conditions characterize a valid stoichiometry vector.

Algorithm `DIOPHANTINE`**.** To turn Eq. (16-17) into an algorithm enumerating the solutions, observe that the $i+1$th Σ consists of finding the stoichiometry vector indices in the range $i+1, \ldots, d$, which accounts for the remaining mass $(M+\varepsilon)_i^-[\mathbf{S}]$. Algorithm 1 presents `DIOPHANTINE`, the corresponding recursive procedure. The recursion tree explored by this algorithm is presented on Fig. 1. Its size, defined as its number of edges, represents the cost of the execution. A leaf is called fertile if it yields a solution, and sterile otherwise. Likewise, an edge is called fertile if it leads to at least one fertile leaf, and sterile otherwise. We note in passing that when the masses are sorted in descending order, one case of output-sensitivity is easily detected. Indeed, if

$$w_p = w_{min} \leq 2 \times \varepsilon, \tag{19}$$

all leaves are fertile—the length of the interval defined in Eq. (14) is 2ε so that is contains at least one solution for s_p.

Regarding the memory footprint, because the stoichiometry vector $\mathbf{S}$ is passed by reference, a unique vector is used along the whole recursion tree, and we have:

Observation. 1. *Algorithm* `DIOPHANTINE` *takes* $\Theta(p)$ *storage.*

Remark 2 (On positive versus non-negative solutions.)**.** *As seen from the sigmas of Eq. (16), algorithm* `DIOPHANTINE` *generates all non-negative solutions. If minimal stoichiometries for the proteins are imposed, compliant solutions can be generated by starting the summations at these stoichiometries—in particular, staring at 1 rather than 0 yields positive solutions. Such solutions can also be generated by subtracting the mass corresponding to these constraints from the target mass, and seeking non-negative solutions.*

Algorithm 1 Algorithm `DIOPHANTINE`**.** The algorithm stores the stoichiometry vectors in a set Sol. The recursive function takes four arguments : the stoichiometry vector $\mathbf{S}$ under construction, passed by reference as indicated by the ampersand & (C++ convention); the remaining mass $m = M_{i-1}^{-}[\mathbf{S}]$ of Eq. (11); the error threshold ε; and the index $i \in [1, \ldots, p]$ of the protein type to be processed by the current recursive call. The weights are sorted in the decreasing order. The initial call is `DIOPHANTINE`$(\mathbf{S_0}, M+\varepsilon, \varepsilon, 1)$, where $\mathbf{S_0}$ is a stoichiometry vector whose p entries are set to 0, M is the target mass, and ε is the allowed error. The sub-routine `keep_value` checks whether the stoichiometry j of the ith protein is admissible—as one may impose a lower and/or upper bound on the stoichiometry of that type.

```
DIOPHANTINE(Stoi_vector & S, unsigned m, unsigned ε, unsigned i)
// Stop the recursion if we are on the last protein type
if i == p then
  q_min = ⌈(m − 2ε)/w_p⌉ // p-th type: min stoichiometry
  q_max = ⌊m/w_p⌋        // p-th type: max stoichiometry
  for j = q_min to q_max do
    if keep_value(p, j) then
      S[p] = j
      Insert S into Sol

  // Recurse to set the remaining stoichiometries
else
  quotient = ⌊m/w_i⌋ // i-th type: max stoichiometry
  for j = 0 to quotient do
    if keep_value(i, j) then
      S[i] = j //Set the i-th stoichiometry
      DIOPHANTINE(S, m − (j w_i), ε, i + 1) // Recursion
```

On the sortedness of weights. The cost of a particular enumeration problem is provided by the number of edges of the recursion tree explored by `DIOPHANTINE`, which actually depends on the sortedness of the vector $\mathbf{W}$ of weights:

Observation. 2. *The size of the recursion tree of algorithm* `DIOPHANTINE` *is minimized when the vector of weights is sorted in descending order.*

Proof. The proof is divided into three steps. First, we prove that swapping two consecutive weights w_i and w_{i+1} only changes the number of node at depth i. Second, that swapping w_i and w_{i+1} results in a smaller number of node at depth i only if after reordering, $w_i \geq w_{i+1}$. Third, that iteratively applying this results leads to the optimal weight reordering $w_1 \geq w_2 \geq \cdots \geq w_p$

Point 1: Consider the upper bound of Eq. (14). Without reordering, a node of the exploration tree at depth $k < p$ is a solution to $\sum_{i=1}^{k} s_i \times w_i \leq M + \varepsilon$, and the number of such node is equal to the corresponding number of such solutions (which we denote here by $\#_k$). For depth p, $\#_p$ is the number of solution as specified by Eq. (17). Thus, the number of tree node is $\#_1 + \#_2 + \cdots + \#_p$. Let now swap w_i and w_{i+1}. The numbers of nodes at depth 1 to $i-1$ do not depend on w_i nor on $w_i + 1$, and thus do not change. $\#_{i'}$, the new number of nodes at depth i is now the number of solutions to $\sum_{j=1}^{i-1}(s_j \times w_j) + s_{i+1} \times w_{i+1} \leq M + \varepsilon$, which is different from $\#_i$. The number of nodes at depth $i+1$ is the number of solutions to $\sum_{j=1}^{i-1}(s_j \times w_j) + s_{i+1} \times w_{i+1} + s_i \times w_i \leq M + \varepsilon$, which is the same as $\#_{i+1}$, and the same holds for larger depths.

Point 2: For any branch defined by a sub-solution $\mathbf{S} = (s_1, ..., s_{i-1})$, the number of nodes at depth i is $\lfloor (M+\varepsilon)_{i-1}^{-}[\mathbf{S}]/w_i \rfloor + 1$. This number is smaller for larger values of w_i, and is thus minimum if we reorder w_i and w_{i+1} so that $w_i \geq w_{i+1}$

Point 3: The optimal weights reordering is then by decreasing order of weights (i.e. when $w_1 \geq w_2 \geq \cdots \geq w_p$). This is proved easily by remarking that using any other weights reordering suppose that there exist a i for which $w_i < w_{i+1}$, implying that the number of nodes in the corresponding exploration trees can be decreased by swapping protein types i and $i+1$. □

3.2 Enumeration Based on Dynamic Programming

The classical solution to the money changing problem based on dynamic programming is well known [BL05a]. It uses a binary table stating whether the target mass is decomposable using a subset of proteins, this table being used by the backtracking algorithm reporting all solutions. In the sequel, we show that a slight modification of the construction of the binary table based on Eq. (15) helps in accommodating the interval case. The enumeration algorithm using this table shall be denoted `DP++`, and is called as DP++$(M + \varepsilon, p)$.

Assume that the weights w_i are sorted by increasing value. Along the course of the backtracking, denote m the mass remaining from $M + \varepsilon$ once a set of protein instances have been used. That is, given the stoichiometry vector $\mathbf{S}$, one has remaining mass,

$$m = M + \varepsilon - \sum_{s_i \in \mathbf{S}} s_i w_i \tag{20}$$

We wish to build a binary table whose semantics is the following:

$$B[i, m] = 1 \quad \Leftrightarrow \quad \exists \text{ a mass } x \text{ in the interval } [max(0, m - 2\epsilon), \ldots, m] \text{ such that } x \text{ is decomposable over } \{w_1, w_2, ...w_i\}.$$

Denote $\#s_1$ the number of non-negative integer values satisfying $\lceil (m-2\varepsilon)/w_1 \rceil \leq s_1 \leq \lfloor m/w_1 \rfloor$. For $i = 1, \ldots, M+\varepsilon$ and $i = 1, \ldots, p$, the binary table $B[i,m]$ as follows:

$$\text{For } i = 1 : B[1,m] = 1 \text{ if } \#s_1 \geq 0, \text{ and 0 otherwise} \tag{21}$$

$$\text{For } i > 1 : B[i,m] = \begin{cases} B[i-1,m] \text{ if } m < a_i \\ B[i-1,m] \vee B[i,m-a_i] \text{ otherwise.} \end{cases} \tag{22}$$

`DP++` consists of backtracking as usual using this binary table. `DP++` is also output sensitive, a property inherited from the original algorithm. The number of backtracking steps for each solution being equal to $(M+\varepsilon)/w_1$ in the worst case, if $D(M,\varepsilon)$ denotes the total number of solutions, the enumeration runs in time $O((M+\varepsilon)\ D(M,\varepsilon)/w_1)$.

3.3 Solution Sketches to Represent a Solution Set

Because an interval SD problem may admit a large solution set $\mathcal{D}$, the question arises to represent all such solutions in a compact way. For a solution $\mathbf{S} \in \mathcal{D}$, we define the stripped solution $\underline{\mathbf{S}}$ by

$$\underline{\mathbf{S}} = \{\ldots, \underline{s_i}, \ldots\} \text{ with } \underline{s_i} = s_i - m_i, \text{ and } m_i = \min_{\mathbf{S} \in \mathcal{D}} s_i. \tag{23}$$

Note that the stoichiometries m_i define the *background* of $\mathcal{D}$. In addition, we define the skeleton $\mathbf{B}$ of a solution $\mathbf{S}$ by

$$\mathbf{B} = \{\ldots, b_i, \ldots\} \text{ with } b_i = 1 \text{ if } \underline{s_i} > 0, \text{ and 0 otherwise.} \tag{24}$$

The set of ones in a skeleton vector identify the protein types $\mathbf{W_B}$ involved in a stripped solution. Since inclusion among such sets defines a partial order, we construct the following Hasse diagram, called the *sketch* of the solution set $\mathcal{D}$:

- A node corresponds to a set $\mathbf{W_B}$. Each such node is associated the solutions of $\mathcal{D}$ whose skeleton defines $\mathbf{W_B}$. The multiplicity μ of the node is the number of such solutions.
- An edge links the nodes $\mathbf{W_B}$ and $\mathbf{W_D}$ iff $\mathbf{W_B} \subset \mathbf{W_D}$, and there does not exist a node $\mathbf{W_C}$ such that $\mathbf{W_B} \subset \mathbf{W_C} \subset \mathbf{W_D}$.

Note that stripping a solution set $\mathcal{D}$ is mandatory for positive solutions of an interval SD problem — or the sketch would involve a single node corresponding to all protein types.

4 Experiments: Implementation and Datasets

In this section, we sketch the software developed, and introduce several datasets used in our experiments, namely `Data-Bio` (10 instances), `Data-Pseudo-Bio10` (200 instances), `Data-Prime10` (200 instances).

4.1 Implementation sketch

Algorithms. Four algorithms were implemented in (generic) C++:

- The algorithm of [BL05b] to compute Frobenius numbers.
- The FPTAS of [GKM$^+$] to estimate the number of solutions of any knapsack problem.

- The algorithms `DIOPHANTINE` and `DP++` from section 3 to solve the interval SD problem.
- The algorithm of [BL05a] to solve SSP, called `DECOMP`. Recall that this algorithm has a pre-processing step consisting of computing the so-called *extended residue table* (ERT), followed by the backtracking step which decomposes a particular mass. To solve the interval SD problem, we compute the ERT once, and call the decomposition procedure for every mass in the interval.

The first three are made available within the program ADDICT, which can be downloaded form `http://team.inria.fr/abs/addict/`.

To avoid overflows, unsigned integers coded on 64 rather than 32 bits were used. Such number types respectively accommodate integers up to $2^{64} - 1 \sim 0.18 \times 10^{20}$ and $2^{32} - 1 \sim 0.42 \times 10^{10}$, and instances with more than 10^{10} solutions are commonly faced—see Experiments. We note that on Linux operating systems, the type `unsigned long long` is always coded on 8 byte, be the system 32 or 64 bits, see e.g. `http://en.cppreference.com/w/cpp/language/types`.

Running times. Practically, the experiments were conducted on a DELL PRECISION T7400 computer equipped with 8Go of RAM, running Fedora Core 14. For all programs, a cut-time of three hours was set, in order to handle the case of enumeration problems yielding an astronomical number of solutions. For such cases, we estimated the number of solutions using the FPTAS strategy of [GKM$^+$]. In fact, whenever the 3 hours time limit was hit, we obtained a lower bound on the number of solutions by taking the maximum value of the number of solutions computed by `DIOPHANTINE` upon hitting the cut-time, and the lower bound computed from the FPTAS.

In analyzing running times, for algorithms `DECOMP` and `DP++`, the total running was split into pre-processing (binary table building), and post-processing (enumerating the solutions), e.g. $t^{\text{Tot}}_{\texttt{DECOMP}} = t^{\text{Pre}}_{\texttt{DECOMP}} + t^{\text{Post}}_{\texttt{DECOMP}}$.

4.2 Biological Dataset

In the following, we briefly present the 10 biological complexes processed, whose masses span the range 321,274 – 5,276,467 Da (Table 1). This table lists experimental molecular weights, except for NPC (Y-ring and single spoke), where theoretical molecular weights were computed from the known stoichiometries. The reader is referred to the supplemental section 8.1 for more details.

Yeast 19S Proteasome lid. Proteasomes are protein assemblies involved in the elimination of damaged or misfolded proteins, and the degradation of short-lived regulatory proteins. The most common form of proteasome is the 26S, which involves two filtering caps (the 19S), each cap involving a peripheral lid, composed of 9 distinct protein types each with unit stoichiometry [STA$^+$06].

COP9 Signalosome. The COP9 Signalosome is a multi-functional complex primarily involved in ubiquitin mediated proteolysis linked to diverse cellular activities such as signal transduction, cell cycle progression and transcriptional regulation. It is composed of 8 protein protein types each with unit stoichiometry.

Eukaryotic Translation factor EIF3. Eukaryotic initiation factors (eIF) are proteins involved in the initiation phase of the eukaryotic translation. They form a complex with the 40S ribosomal subunit, initiating the ribosomal scanning of mRNA. Among them, EIF3 consists of 13 different protein types each with unit stoichiometry.

Yeast Exosome. The exosome is a 3'- 5' exonuclease complex involved in RNA processing and degradation. The yeast exosome is composed of 10 different protein types with unit stoichiometry [HDT$^+$06].

Rotary ATPases. Rotary ATPases are membrane associated molecular machines involved in energy conversion by coupling ATP hydrolysis (or synthesis) with proton (or Na^+) translocation across biological membranes. Practically, we investigate the intact TtATPase, and four sub-complexes of EhATPase.

Yeast Nuclear Pore Complex (NPC). The NPC, which is the largest protein assembly known to date in the eukaryotic cell [WR10, DH08], is a protein assembly anchored in the nuclear envelope, regulating the nucleo-cytoplasmic transport. It is composed of ~ 30 distinct proteins types each present in multiple copies. It has eight-fold radial symmetry and consists of eight spokes, each containing 57 protein instances. One particular complex, the Y-complex, involving 7 protein types with unit stoichiometry, is present in two sets of 8 copies. Each such set presumably forms one ring (the Y-ring) involving 56 protein instances [FMPS$^+$12, DDC12].

4.3 Artificial Datasets

Prime numbers instances: the `Data-Prime10` dataset. The use of prime number weights prevents relative multiplicity of individual weights, and therefore avoids outright hardness of the instance stemming from so-called spanner sets [Pis05]. To generate the kth synthetic complex, with $k \in 1, \ldots, 10$, p prime numbers were chosen randomly from the list of prime numbers from 7,000 to 70,000, in accordance with the weights (in Daltons) of individual proteins involved in the biological complexes of `Data-Bio`. Moreover, 20 instances for each kth weight vector(W_k) were generated by using 20 target masses uniformly spaced in the range

$$[\sum w_i, 2g_0^+(\mathbf{W_k})]. \tag{25}$$

These 200 instances are referred to as `Data-Prime10-i-j`.

Random biological instances: the `Data-Pseudo-Bio10` dataset. To generate synthetic complexes using the masses of real proteins, we replaced the set of prime numbers by the masses of ~ 6700 YEAST proteins retrieved from the non-redundant UNIPROT protein database, see `http://www.uniprot.org/help/about`. The theoretical molecular weight of proteins were calculated by adding up the weight of constituent amino acids in the proteins. (To be consistent with `Data-Prime10`, only masses beyond 7 kDa were retained.) As above, 10 vectors of masses were picked, with 20 target masses in the interval of Eq. (25), yielding 200 instances denoted `Data-Pseudo-Bio10-i-j`.

5 Experimental Results

The following comments are in order. First, in structural proteomics, the protein types involved in a given protein complex can generally be determined by mass spectrometry upon dismantling the complex into its individual sub-units, thanks to denaturing conditions. This explains why Table 1 is about positive solutions—rather than non-negative solutions. But all the remaining experiments have been conducted with non negative solutions. Second, when three series of numbers are reported, they refer to the three datasets `Data-Bio`, `Data-Pseudo-Bio10` and

`Data-Prime10`, enumerated in this order. Third, three error levels typical in structural proteomics were used, namely 0%, 0.1% and 1% — the value ε of Eq. (2) being set to the error level times the target mass.

Notation-wise, the median of a list L of values is denoted median(L). Also, the linear correlation coefficient (i.e. Pearson) between two variables X and Y is denoted Corr(X, Y).

5.1 Biological Examples: Enumeration Matters Even at Null Noise Level

The formula recalled in Eq. (4) shows an asymptotic polynomial growth of the number of solutions, yet, are situations with multiple solutions commonplace in biology? Table 1 answers this question positively, with multiple solutions even at null noise level in some cases. The case of the NPC is interesting. The case of one full spoke shows that the stoichiometry determination problem in this case is ill-posed, with an astronomical number of solutions. The same holds for the NPC-Y-ring system, which consists of 8 copies of the Y-complex, since a total of 788 solutions are obtained.

The last column of this table also shows that a large value M/g_0^+ is a good hint at a large number of solutions.

Also, a solution sketch for positive solutions, as defined in section 3.3, is presented on Fig. 2 for the system EhATPase-sub-4. While on this simple example every node of the Hasse diagram corresponds to a single solution, other example yield more complex (and cluttered) Hasse diagrams, with numerous solutions per node. Interestingly, we also observed that for non-negative solutions and even a 0.1% noise level, all solution sketches of biological complexes pretty much involve all possible tuples (supplemental Figs. 17 and 18). That is, the Hasse diagram essentially contains all $\sum_{k=1}^{p} \binom{p}{k}$ nodes which are defined by Eq. 24.

5.2 Counting Solutions and Convergence to the Denumerant

Rationale. As noticed in section 2, counting solutions ahead of computing them is of interest to avoid generating a large number of (useless) solutions. We compared the accuracy of the estimations provided in [Pad71, Bd72, Lam74] against those of the FPTAS algorithm of [GKM$^+$].

One comment is also in order about *hard instances*, i.e. instances such that `DIOPHANTINE` does not terminate within a prescribed time limit—three hours in this work. According to [Pis05], two parameters lead to hard UKP and SSP instances: the magnitude of weights, and the linear redundancy between the weights. Intuitively, if a given weight is a linear combination of other weights, additional solutions can be generated. However, beyond the Frobenius number and in the asymptotic regime, the number of solutions of SSP is given by Eq. 4. Therefore, instead of using a target mass proportional to the sum of masses, as in [Pis05], we systematically explored ranges of target masses expressed in units of Frobenius number.

Results: counting solutions. The biological examples just discussed show that estimating the number of solutions is important to detect ill posed problems.

Table 2 compares various strategies to estimate the number of solutions to a SD problem. First, we plugged the upper and lower bounds provided by [Pad71], [Bd72], and [Lam74] into Eqs. (9-10). For each biological system, we retained the best bounds obtained—those yielding the tighter interval (Table 2, columns 2-4). Second, we also computed estimates using the FPTAS of [GKM$^+$] (Table 2, columns 5-7).

On the one hand, the combinatorial bounds are not of real interest, since at least two orders of magnitude of difference between $\underline{\#}\mathrm{UKP}(p, \mathbf{W}, M)$ and $\overline{\#}\mathrm{UKP}(p, \mathbf{W}, M)$, yielding a null lower bound. On the other hand, the bounds from the FPTAS are much tighter.

Results: convergence towards the denumerant. Due to the rapid rise of the number of solutions, an interesting problem is the speed of the convergence of the number of solutions of a SD problem to the denumerant.

To study this convergence, a toy dataset called `Data-Quad4` consisting of the quadruplet $\{10, 15, 32, 48\}$ from [SS11] was used. A total of 100 instances uniformly distributed in the range $[\sum w_i, 41g_0]$ was used, the jth instance being denoted `Data-Quad4` $- j$. We also studied this convergence for a real biological system of Yeast Exosome complex. A total of 40 instances having target masses uniformly distributed in the range $[\sum w_i, 20g_0]$. The results on this system and on a real biological system are presented on Figs. 3 and 4, and show a rapid convergence to the denumerant. The ratio of denumerant to the number of non-negative solutions grows to 96% at $40g_0$ starting from 29%, in case of `Data-Quad4` whereas, for the real biological instance, this ratio grows to 70% at $10g_0$ from its initial value of 12%. The Yeast Exosome instances having target mass greater than $11g_0$ did not terminate within the time limit of 3h. Note that to compare with Denumerant, non-negative solutions are plotted.

5.3 Running Times: `DECOMP`, `DIOPHANTINE`, `DP++`

Rationale. To compare the three running times $t^{\mathrm{Tot}}_{\texttt{DIO}}, t^{\mathrm{Tot}}_{\texttt{DECOMP}}, t^{\mathrm{Tot}}_{\texttt{DP++}}$, we processed our three datasets at the noise levels 0%, 0.1%, and 1% (Fig. 5).

Results. While `DIOPHANTINE` and `DP++` completed all instances within the imparted time, regardless of the noise level, on a per-dataset basis, `DECOMP` finished on 10/10, 200/200, 200/200 cases at zero noise level, 9/10, 104/200, 93/200 cases at 0.1% noise level, and 8/10, 56/200, 51/200 cases at 1% noise level.

At null noise level, one gets $\mathrm{median}(\{t^{\mathrm{Tot}}_{\texttt{DECOMP}}/t^{\mathrm{Tot}}_{\texttt{DIO}}\}) = 1.4$, $\mathrm{median}(\{t^{\mathrm{Tot}}_{\texttt{DIO}}/t^{\mathrm{Tot}}_{\texttt{DP++}}\}) = 13.66$, $\mathrm{median}(\{t^{\mathrm{Tot}}_{\texttt{DECOMP}}/t^{\mathrm{Tot}}_{\texttt{DP++}}\}) = 16.02$. That is, `DP++` is faster than `DECOMP` and `DIOPHANTINE` by about one order of magnitude. It should be noticed in particular that the better performance of `DP++` over `DECOMP` owes to the constants involved in the output-sensitive complexity, which are respectively of $O(D(M,0)\ M/w_{min})$ and $O(D(M,0)\ p\ w_{min})$. Practically, biological complexes are such that $p \sim 10, w_{min} \sim 10$ kDa, and $M \sim 1$ MDa.

At 0.1% and 1% noise level, `DECOMP` is not competitive anymore, since it is outperformed by three to four orders of magnitude. Clearly, the wider the interval $[M - \varepsilon, \ldots, M + \varepsilon]$, the more challenging the instances for `DECOMP`, an issue precisely analyzed in section 5.5.

In presence of noise, the two competitors are thus `DIOPHANTINE` and `DP++`. At 0.1% and 1% respectively, one has $\mathrm{median}(\{t^{\mathrm{Tot}}_{\texttt{DP++}}/t^{\mathrm{Tot}}_{\texttt{DIO}}\}) = 1.07$, $\mathrm{median}(\{t^{\mathrm{Tot}}_{\texttt{DP++}}/t^{\mathrm{Tot}}_{\texttt{DIO}}\}) = 1.2$. More interestingly, `DIOPHANTINE` is never twice slower than `DP++`, while in the worst case, `DP++` is 17 (resp. 20) times slower than `DIOPHANTINE` at 0.1% (resp. 1%). Thus, `DIOPHANTINE` is more stable. We also notice that asymptotically when the target mass increases, the ratio of running times converges to one, hinting at an output sensitive behavior of `DIOPHANTINE`— recall that `DP++` is output sensitive.

5.4 Output sensitivity of `DIOPHANTINE`

Rationale. In the exact case, `DIOPHANTINE` can only be output sensitive for a target mass beyond the Frobenius number, since any call with a non representable mass necessarily results in

useless operations. The running times of DIOPHANTINE being comparable to those of the output sensitive algorithm DP++, we analysis the output sensitive behavior of DIOPHANTINE.

Results: running time versus number of non-negative solutions. We first computed the Pearson correlation coefficient between $t^{\text{Tot}}_{\texttt{DIO}}$ and the number of solutions (Fig. 6). With values equal to 0.97, 0.89 and 0.97 for the three datasets at 0.1% noise level, and to 0.97, 0.89 and 0.97 at 1% noise level, this correlation is excellent.

Results: running time versus the recursion tree size. The correlation between $t^{\text{Tot}}_{\texttt{DIO}}$ and the recursion tree size is even better (Fig. 7), with coefficients of 0.99, 0.99 and 0.99 at 0.1% noise level, and of 0.99, 0.95 and 0.98 at 1% noise level, respectively for the three datasets.

We further plotted the proportion of fertile edges as a function of the target mass (Fig. 8). Note that this proportion is 1 when the condition of Eq. (19) holds, which is the case for 0/410 and 338/410 instances at 0.1% and 1% respectively. As expected, this proportion increases with the error level: at 1%, all the instances which do not meet the condition of Eq. (19) have a ratio above 0.6, while no instance has such a high ratio at 0.1%.

5.5 Decreasing Performances of DECOMP upon Increasing the Noise Level

Rationale. The larger the noise level, the better the performances of DP++ w.r.t. DECOMP. Also, algorithm DP++ is output sensitive for the interval SD problem, while algorithm DECOMP is output sensitive for the exact SD problem but possibly not for the interval SD problem. We wish to relate these two facts, using two ingredients:

- First, the decomposition of the running time into pre and post-processing, namely $t^{\text{Tot}}_{\texttt{DP++}} = t^{\text{Pre}}_{\texttt{DP++}} + t^{\text{Post}}_{\texttt{DP++}}$, and likewise for DECOMP.
- Second, the number of nodes explored by DP++ and DECOMP. For DP++, this number is the number of nodes in the backtracking procedure, as explained in section 3.2. For DECOMP, this number is equal to the sum of the number of explored nodes for target masses in the range $[M - \varepsilon, M + \varepsilon]$.

Results: for algorithm DP++, the post-processing time $t^{\text{Post}}_{\texttt{DP++}}$ converges to total runtime $t^{\text{Tot}}_{\texttt{DP++}}$ when the number of solutions increases. We first analyze the relative importance of the pre-processing and post-processing times for both algorithms. As shown by the scatter plots (Fig.9), the post-processing step always dominates for DECOMP, while for DP++ it becomes dominant as the number of solutions increases. For the latter algorithm, the linear trend between $t^{\text{Post}}_{\texttt{DP++}}/t^{\text{Pre}}_{\texttt{DP++}}$ and the number of nodes but also the number of solutions is confirmed by correlation coefficients beyond 0.9, at both noise levels 0.1% and 1% (Table 3, and supplemental Figs. SI-11, SI-12). Interestingly, the correlation between the ratio of running times $t^{\text{Post}}_{\texttt{DP++}}/t^{\text{Pre}}_{\texttt{DP++}}$ and the target mass is poor, with coefficients of 0.32 and 0.11 at 0.1% and 1% noise level respectively.

Results: the decreasing performance of DECOMP owes to a redundant post-processing. Since algorithm DP++ is output sensitive, we take it as a yardstick and postulate that that if the number of nodes explored by DECOMP incurs an x-fold increase with respect to the number of nodes explored by DP++, so should the running-times. Phrased differently, we compute the correlation coefficient

$$\text{Corr}(\frac{t^{\text{Post}}_{\texttt{DECOMP}}}{t^{\text{Post}}_{\texttt{DP++}}},\ \frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}). \tag{26}$$

As seen from Table 4, this first correlation is strong, with values of 0.75 and 0.79 at noise levels of 0.1% and 1% respectively. On removing two outliers at each of the noise level, the coefficients improve to 0.86 and 0.93, respectively (Supplemental Fig. SI-14). The linear relationship between the two terms of the correlation defined by Eq. (26), together with the convergence of $t^{\text{Post}}_{\texttt{DECOMP}}$ to $t^{\text{Tot}}_{\texttt{DECOMP}}$, and the convergence of $t^{\text{Post}}_{\texttt{DP++}}$ to $t^{\text{Tot}}_{\texttt{DP++}}$ establish that the increased total running time $t^{\text{Tot}}_{\texttt{DECOMP}}$ linearly depends on the number of nodes visited by DECOMP. In particular, taking as reference $\#nodes_{\text{DP++}}$ since algorithm DP++ is output sensitive, the ratio $\#nodes_{\text{DECOMP}}/\#nodes_{\text{DP++}}$ measures the redundant work of DECOMP corresponding to the successive calls to solve the individual SD problems.

Since the previous line of argument holds asymptotically, namely when $t^{\text{Post}}_{\texttt{DP++}}$ converges to $t^{\text{Tot}}_{\texttt{DP++}}$, we also investigate directly

$$\text{Corr}(\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}},\ \frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}). \tag{27}$$

With values of 0.48 and 0.35 at noise levels 0.1% and 1%, this correlation is weak. The problem

in computing the coefficient of Eq. (27) with $N = 203$ [3] and with $N = 112$ [4] instances respectively at 0.1% and 1% noise levels is that the overall correlation is spoiled by the instances for which the pre-processing time still dominates in a run of DP++. To measure the incidence of the convergence of $t^{\text{Post}}_{\texttt{DP++}}$ to $t^{\text{Tot}}_{\texttt{DP++}}$ when the number of solution increases, we therefore resort to a sequence of correlation coefficients, the set of instances used to investigate the correlation corresponding to the cases where the ratio $t^{\text{Post}}_{\texttt{DP++}}/t^{\text{Pre}}_{\texttt{DP++}}$ is larger than some threshold. More precisely, given a set of N instances, we compute N correlation coefficients as follows:

- Based on the running times of algorithm `DP++` we compute

$$\alpha_i = t^{\text{Post}}_{\texttt{DP++}}/t^{\text{Pre}}_{\texttt{DP++}} \tag{28}$$

  for each instance, and sort the N instances by increasing α_i value.

- For each index $i = 1, \ldots, N-1$, let

$$S_{>i} : \text{ the set of instances such that } \alpha_j > \alpha_i. \tag{29}$$

  We compute the Pearson correlation coefficient $\text{Corr}(\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}}, \frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}})$ on the set $S_{>i}$.

The plot of these coefficients is presented on Fig. 10. If one omits the last 10 coefficients — which are inherently unstable since they involve less than 10 points, the values obtained for the coefficient of Eq. (27) now reach 0.9. Two points are noticeable. First, at 0.1% noise level, the correlation rises rapidly as a function of the α_i value. As scatter plots show (supplemental Fig. 19), the decreasing section for the trailing 30 instances or so owes to the paucity of the dataset. Second, at 1% noise level, the correlation rises monotonically over the range of α_i values explored. This indicates that the increase in number of nodes of `DECOMP` translates directly on the running time $t^{\text{Tot}}_{\texttt{DECOMP}}$, which is expected since when α_i increases, the post-processing time which depends on the number of nodes converges to the total running time. Again, the ratio $\#nodes_{\text{DECOMP}}/\#nodes_{\text{DP++}}$ directly measures the redundancy of the work carried out by algorithm `DECOMP` with respect to the output sensitive algorithm `DP++`, this redundancy having a linear incidence on the running time.

Discussion. These observations are actually consistent with the complexities of the pre-processing steps and the number of nodes generated. Indeed, the pre-processing step in the case of `DECOMP` is independent of the target mass of the instance ($O(p\ w_{min})$), while in case of `DP++` it is linearly dependent on the target mass ($O((M+\epsilon)p)$).

As for the number of nodes generated during the post-processing step, the degraded performances of `DECOMP` for the interval SD problem is actually expected. Indeed, while a given stoichiometry vector is generated exactly once for any interval SD problem (the solutions to two different target masses are different), redundant calculations are possibly carried out by different runs of `DECOMP` for consecutive target masses. Ideally, one would measure the redundancy of calculations carried out to generate two different solutions is precisely measured by the longest common sub-sequence between the stoichiometry vectors of these two solutions. More plainly, we use the number of steps required by the backtracking procedure.

[3] number of instances for which both the algorithms terminated within 3h with 0.1% noise level. Note that `DP++` terminated for all 410 instances.

[4] number of instances for which both the algorithms terminated within 3h with 1% noise level. Note that `DP++` terminated for all 410 instances.

6 Conclusion and Outlook

Mass spectrometry is playing an increasing role to investigate large macro-molecular assemblies, thus complementing more classical techniques such as X ray crystallography or nuclear magnetic resonance, which are better suited for smaller complexes. In this context, the stoichiometry determination (SD) problem, which consists of determining how many copies of each sub-unit are required to account for the observed mass, is the first one to be addressed, before investigating the geometry of the contacts between these sub-units. In this work, we develop a constant memory space enumeration algorithm (`DIOPHANTINE`), and an output sensitive dynamic programming based algorithm (`DP++`), which are shown to outperform state-of-the-art SD algorithms by several (three to four) orders of magnitude. These two algorithms exhibit comparable performances for typical noise levels in the range 0.1% to 1%, which is remarkable since `DIOPHANTINE` does not use any pre-processing and has constant memory footprint. Both algorithms performed to satisfaction on all the biological cases processed, which are the most challenging ones currently investigated in structural biology. It is noticeable that enumeration does matter, since systems with numerous solutions are observed at a moderate (sometimes even null) noise level. Also, the coupling between `DIOPHANTINE` (or `DP++`) and the fully polynomial-time approximation scheme (FPTAS) of [GKM$^+$] provides a powerful way to get a lower bound on the number of solutions, and thus to identify ill-posed SD problems—which admit a too large number of solutions.

From a theoretical standpoint, several outstanding questions remain open. The first one is related to the output sensitivity of `DIOPHANTINE`, which is observed in practice, but cannot be guaranteed in general. Getting finer insights on this phenomenon may allow developing more efficient enumeration schemes—by pruning sterile portions of the recursion tree. The second one is reminiscent from phase transitions, a well known phenomenon for selected hard optimization problems—such as 3-SAT for example. While the SD problem is about enumeration rather than optimization, linear redundancies within protein masses (one can trade a set of proteins against another set) have a direct impact on the rise of the number of solutions. A challenge is therefore to gain insights on the role of linear relationships and the total number of solutions. The third one is the asymptotic analysis of the number of solution of an interval stoichiometry determination problem. While the asymptotics of the denumerant $D(M, 0)$ is known, working out the behavior or $D(M, \varepsilon)$ in conjunction with the distribution of solutions per node of the Hasse of the solution diagram sketch would provide valuable information on solutions as a function of the input masses and the noise level.

References

[ACL$^+$02] P. Aloy, F.D. Ciccarelli, C. Leutwein, A.C. Gavin, G. Superti-Furga, P. Bork, B. Böttcher, and R.B. Russell. A complex prediction: three-dimensional model of the yeast exosome. *EMBO reports*, 3(7):628–635, 2002.

[Agn02] G. Agnarsson. On the sylvester denumerants for general restricted partitions. *Congressus numerantium*, pages 49–60, 2002.

[Alf05] J.L. Ramírez Alfonsín. *The Diophantine Frobenius Problem.* Oxford University Press, 2005.

[Bd72] A.G. Beged-dov. Lower and upper bounds for the number of lattice points in a simplex. *SIAM Journal on Applied Mathematics*, 22(1):106–108, 1972.

[BDF+10] A. Brown, E. Dannenberg, J. Fox, J. Hanna, K. Keck, A. Moore, Z. Robbins, B. Samples, and J. Stankewicz. On a generalization of the frobenius number. *Journal of Integer Sequences*, 13(2):3, 2010.

[Bel43] E.T. Bell. Interpolated denumerants and lambert series. *American Journal of Mathematics*, 65(3):382–386, 1943.

[Bel57] R.E. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[BL05a] S. Bocker and Z. Liptak. Efficient mass decomposition. In ACM, editor, *ACM Symposium on Applied Computing*, 2005.

[BL05b] S. Böcker and Z. Lipták. The money changing problem revisited: Computing the frobenius number in time $o(ka1)^*$. In Lusheng Wang, editor, *Computing and Combinatorics*, volume 3595 of *Lecture Notes in Computer Science*, pages 965–974. Springer Berlin / Heidelberg, 2005.

[BLM+08] Sebastian Böcker, Zsuzsanna Lipták, Marcel Martin, Anton Pervukhin, and Henner Sudek. DECOMP—from interpreting mass spectrometry peaks to solving the Money Changing Problem. *Bioinformatics*, 24(4):591–593, 2008.

[BR11] J.L.P. Benesch and B.T. Ruotolo. Mass spectrometry: come of age for structural and dynamical biology. *Current opinion in structural biology*, 21:641–649, 2011.

[Com74] L. Comtet. *Advanced Combinatorics: The art of finite and infinite expansions*. Not Avail, 1974.

[DDC12] T. Dreyfus, V. Doye, and F. Cazals. Assessing the reconstruction of macro-molecular assemblies with toleranced models. *Proteins: structure, function, and bioinformatics*, 80(9):2125–2136, 2012.

[DF80] R.G. Dromey and G.T. Foyster. Calculation of elemental compositions from high resolution mass spectral data. *Analytical Chemistry*, 52(3):394–398, 1980.

[DFZ+07] E. Damoc, C.S. Fraser, M. Zhou, H. Videler, G.L. Mayeur, J.W.B. Hershey, J.A. Doudna, C.V. Robinson, and J.A. Leary. Structural characterization of the human eukaryotic initiation factor 3 protein complex by mass spectrometry. *Molecular & Cellular Proteomics*, 6(7):1135–1146, 2007.

[DH08] M.A. D'Angelo and M.W. Hetzer. Structure, Dynamics and Function of Nuclear Pore Complexes. *Trends Cell Biology*, 18:456–522, 2008.

[ea01] O. Puig et al. The tandem affinity purification method: A general procedure of protein complex purification. *Methods*, 24:218–229, 2001.

[FMPS+12] J. Fernandez-Martinez, J. Phillips, M.D. Sekedat, R. Diaz-Avalos, J. Velazquez-Muriel, J.D. Franke, R. Williams, D.L. Stokes, B.T. Chait, A. Sali, and M.P. Rout. Structure–function mapping of a heptameric module in the nuclear pore complex. *The Journal of Cell Biology*, 2012.

[FS09] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. Cambridge Univ Pr, 2009.

[GKM+] P. Gopalan, A. Klivans, R. Meka, D. Stefankovic, S. Vempala, and E. Vigoda. An FPTAS for# knapsack and related counting problems. In *IEEE Foundations of Computer Science*.

[HDT+06] H. Hernández, A. Dziembowski, T. Taverner, B. Séraphin, and C.V. Robinson. Subunit architecture of multimeric complexes isolated directly from cells. *EMBO reports*, 7(6):605–610, 2006.

[Jan07] J. Janin. structural genomics: winning the second half of the game. *Structure*, 15:1347–1349, 2007.

[JHP10] R.J. Jackson, C.U.T. Hellen, and T.V. Pestova. The mechanism of eukaryotic translation initiation and principles of its regulation. *Nature Reviews Molecular Cell Biology*, 11(2):113–127, 2010.

[Kar72] R.M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations.*, 6:85–103, 06 1972.

[Kel04] N.L. Kelleher. Peer reviewed: Top-down proteomics. *Analytical chemistry*, 76(11):196–203, 2004.

[Lam74] T.A. Lambe. Bounds on the number of feasible solutions to a knapsack problem. *SIAM Journal on Applied Mathematics*, 26(2):302–305, 1974.

[MHFH10] R. Mahmoudvand, H. Hassani, A. Farzaneh, and G. Howell. The exact number of nonnegative integer solutions for a linear diophantine inequality. *IAENG International Journal of Applied Mathematics*, 40(1), 2010.

[MR12] N. Morgner and C.V. Robinson. Massign: An assignment strategy for maximising information from the mass spectra of heterogeneous protein assemblies. *Analytical Chemistry*, 84(6):2939–2948, 2012.

[Pad71] M.W. Padberg. A remark on "an inequality for the number of lattice points in a simplex". *SIAM Journal on Applied Mathematics*, 20(4):638–641, 1971.

[Pis05] D. Pisinger. Where are the hard knapsack problems? *Computers and Operations research*, 32:2271–2284, 2005.

[PKP04] U. Pferschy, H. Kellerer, and D. Pisinger. *Knapsack Problems.* Springer, 2004.

[PYA09] V. Poirriez, N. Yanev, and R. Andonov. A hybrid algorithm for the unbounded knapsack problem. *Discrete Optimization*, 6(1):110 – 124, 2009.

[RA96] J. L. Ramírez-Alfonsín. Complexity of the frobenius problem. *Combinatorica*, 16:143–147, 1996. 10.1007/BF01300131.

[RDD+12] R.J. Rose, E. Damoc, E. Denisov, A. Makarov, and A.J.R. Heck. High-sensitivity orbitrap mass analysis of intact macromolecular assemblies. *Nature Methods*, 9(11):1084–1086, 2012.

[Sma98] N.P. Smart. *The algorithmic resolution of Diophantine equations*, volume 41. Cambridge Univ Pr, 1998.

[SMBE+09] M. Sharon, H. Mao, E. Boeri Erba, E. Stephens, N. Zheng, and C.V. Robinson. Symmetrical modularity of the cop9 signalosome complex suggests its multifunctionality. *Structure*, 17(1):31–40, 2009.

[SR07] M. Sharon and C.V. Robinson. The role of mass spectrometry in structure elucidation of dynamic protein complexes. *Annu. Rev. Biochem.*, 76:167–193, 2007.

[SS11] J. Shallit and J. Stankewicz. Unbounded discrepancy in frobenius numbers. *Integers*, 11:1–8, 2011.

[STA+06] M. Sharon, T. Taverner, X.I. Ambroggio, R.J. Deshaies, and C.V. Robinson. Structural organization of the 19s proteasome lid: insights from ms of intact complexes. *PLoS biology*, 4(8):e267, 2006.

[THS+08] T. Taverner, H. Hernández, M. Sharon, B.T. Ruotolo, D. Matak-Vinkovic, D. Devos, R.B. Russell, and C.V. Robinson. Subunit architecture of intact protein complexes from mass spectrometry and homology modeling. *Accounts of chemical research*, 41(5):617–627, 2008.

[WR10] S.R. Wente and M.P. Rout. The Nuclear Pore Complex and Nuclear Transport. *Colde Spring Harbor Perspectives in Biology*, 2(10):a000562, 2010.

[ZCGB13] H. Zhang, W. Cui, M.L. Gross, and R.E. Blankenship. Native mass spectrometry of photosynthetic pigment-protein complexes. *FEBS Letters*, 2013.

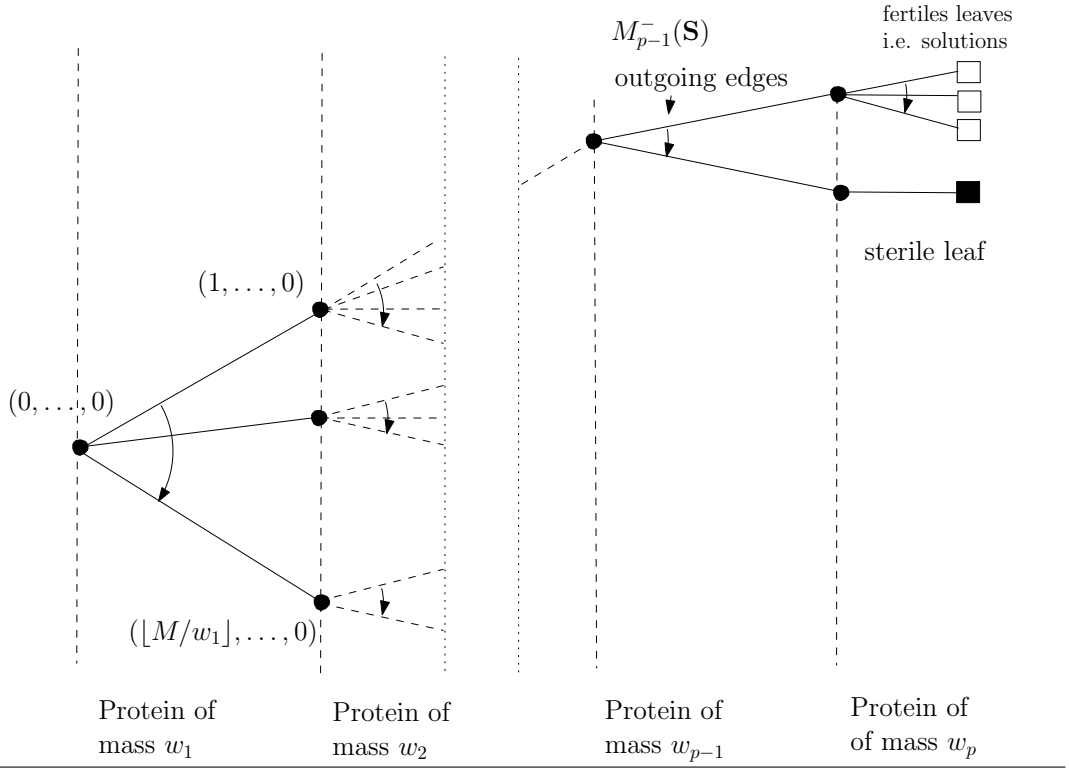
[ZMB+11] M. Zhou, N. Morgner, N.P. Barrera, A. Politis, S.C. Isaacson, D. Matak-Vinković, T. Murata, R.A. Bernal, D. Stock, and C.V. Robinson. Mass spectrometry of intact v-type atpases reveals bound lipids and the effects of nucleotide binding. *Science*, 334(6054):380–385, 2011.

[ZSF+08] Min Zhou, Alan M. Sandercock, Christopher S. Fraser, Gabriela Ridlova, Elaine Stephens, Matthew R. Schenauer, Theresa Yokoi-Fong, Daniel Barsky, Julie A. Leary, John W. Hershey, Jennifer A. Doudna, and Carol V. Robinson. Mass spectrometry reveals modularity and a complete subunit interaction map of the eukaryotic translation factor eif3. *Proceedings of the National Academy of Sciences*, 105(47):18139–18144, 2008.

7 Artwork

7.1 Algorithms

Figure 1 The recursion tree explored by algorithm `DIOPHANTINE`**.** A leaf is called fertile if Eq. (17) has at least one solution, and sterile otherwise. Similarly, any edge of the tree is termed fertile if at least one fertile leaf is found downstream the tree, and sterile otherwise. The size of the recursion tree is defined as its number of edges.

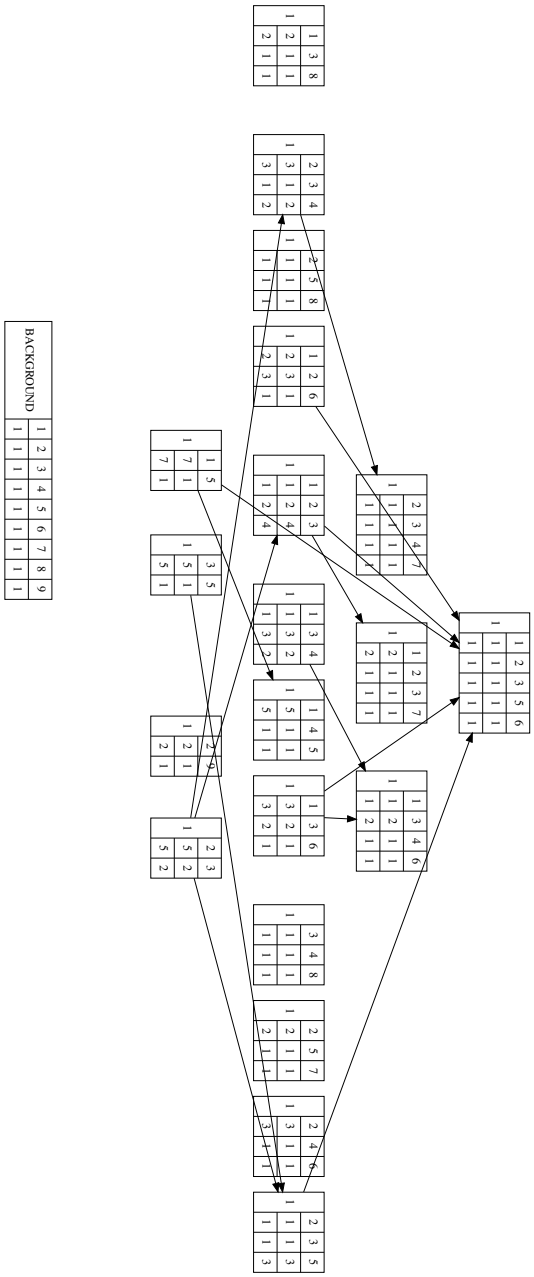


7.2 Stoichiometry Determination for Biological Examples: Enumeration Matters

Table 1 Biological instances can lead to multiple positive solutions, even with null noise level. For the NPC-1-spoke, lower and upper bounds were obtained using UKP-FPTAS of [GKM$^+$] with an error factor of $\delta = 0.1$.

Complex	#sol, 0% noise	#sol, 0.1% noise	#sol, 1% noise	M (Da)	g_0^+	$\#sol(1\%\, noise)/2\varepsilon$	M/g_0^+
COP9	0	1	1	321,274 ± 35	961,855	3.12×10^{-4}	0.33
Y-19S-lid	0	0	1	376,151 ± 369	921,712	1.87×10^{-4}	0.41
EhATPase-sub-5	0	4	39	387,356 ± 230	682,901	5.03×10^{-3}	0.57
Y-exosome	0	13	149	402,708 ± 68	649,185	1.85×10^{-2}	0.62
EhATPase-sub-4	0	20	190	424,441 ± 148	682,901	2.38×10^{-2}	0.62
EhATPase-sub-3	0	74	700	461,674 ± 324	682,901	8.10×10^{-2}	0.68
EhATPase-sub-2	0	224	2,213	500,178 ± 294	682,901	2.32×10^{-1}	0.73
TtATPase	21	24,487	246,242	659,202 ± 131	607,304	18.7	1.08
EIF3	0	0	1	797,999 ± 180	1,257,629	8.05×10^{-5}	0.63
NPC-Y-ring	788	6,900,664	69,042,257	4,603,280	2,282,543	750	2.02
NPC-1-spoke	$[1.72 \times 10^{16}$, $1.73 \times 10^{16}]$	$[1.72 \times 10^{16}$, $4.77 \times 10^{16}]$	$[2.71 \times 10^{17}$, $3.44 \times 10^{17}]$	5,276,467	3,169,210	$\geq 2.57 \times 10^{12}$	1.66

Figure 2 The 20 solutions for EhATPase-sub-4 at 0.1% noise level, see also Table 1 The Hasse diagram presents the sketch of the solution set, as defined in section 3.3, while the background presents the min stoichiometry of each type present in any solution. Each node of the Hasse diagram reads as follows: the left hand side presents the number of solutions. Each row on the right hand side reads as follows: (top) list of protein types (middle, bottom) min and max stoichiometries of all solutions using these types. In the block for *Background information*, first row corresponds to the index of protein types and the second row corresponds to the minimum stoichiometry of these protein types across all the positive solutions. Note in particular that an arrow represents the inclusion between the protein types of two nodes.



7.3 Counting Solutions and Convergence to the Denumerant

Table 2 Estimating the number of positive solutions for five biological systems. Columns 2 to 4: upper and lower bounds obtained by plugging the upper and lower bounds of [Pad71], [Bd72], and [Lam74] into Eqs. (9-10). Columns 5 to 7: upper and lower bounds on the number of positive solutions obtained using the FPTAS of [GKM+], with $\delta = 0.1$.

Complex	# solutions	$M \pm 1\%$		$M \pm 1\%$	
		$\underline{\#}SD$	$\overline{\#}SD$	$\underline{\#}SD$	$\overline{\#}SD$
COP9	1	0	8	1	1
Y19-lid	1	0	10	1	1
Y-exosome	149	0	486	101	183
EIF3	1	0	14	1	1
NPC-1-Spoke	Unknown	0	1.06×10^{22}	2.71×10^{17}	3.44×10^{17}

Figure 3 Number of non-negative solutions vs. asymptototic behavior of the denumerant (Eq. (4)) for the quadruplet $\mathbf{W} = \{10, 15, 32, 48\}$. The target mass is expressed in units of the Frobenius number. *Note that Y-axis is drawn with a logarithmic scale.*

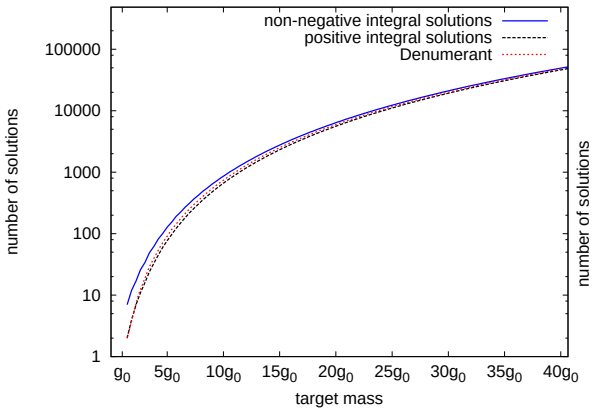
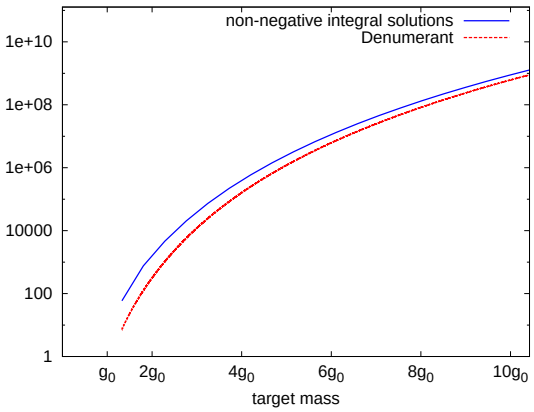
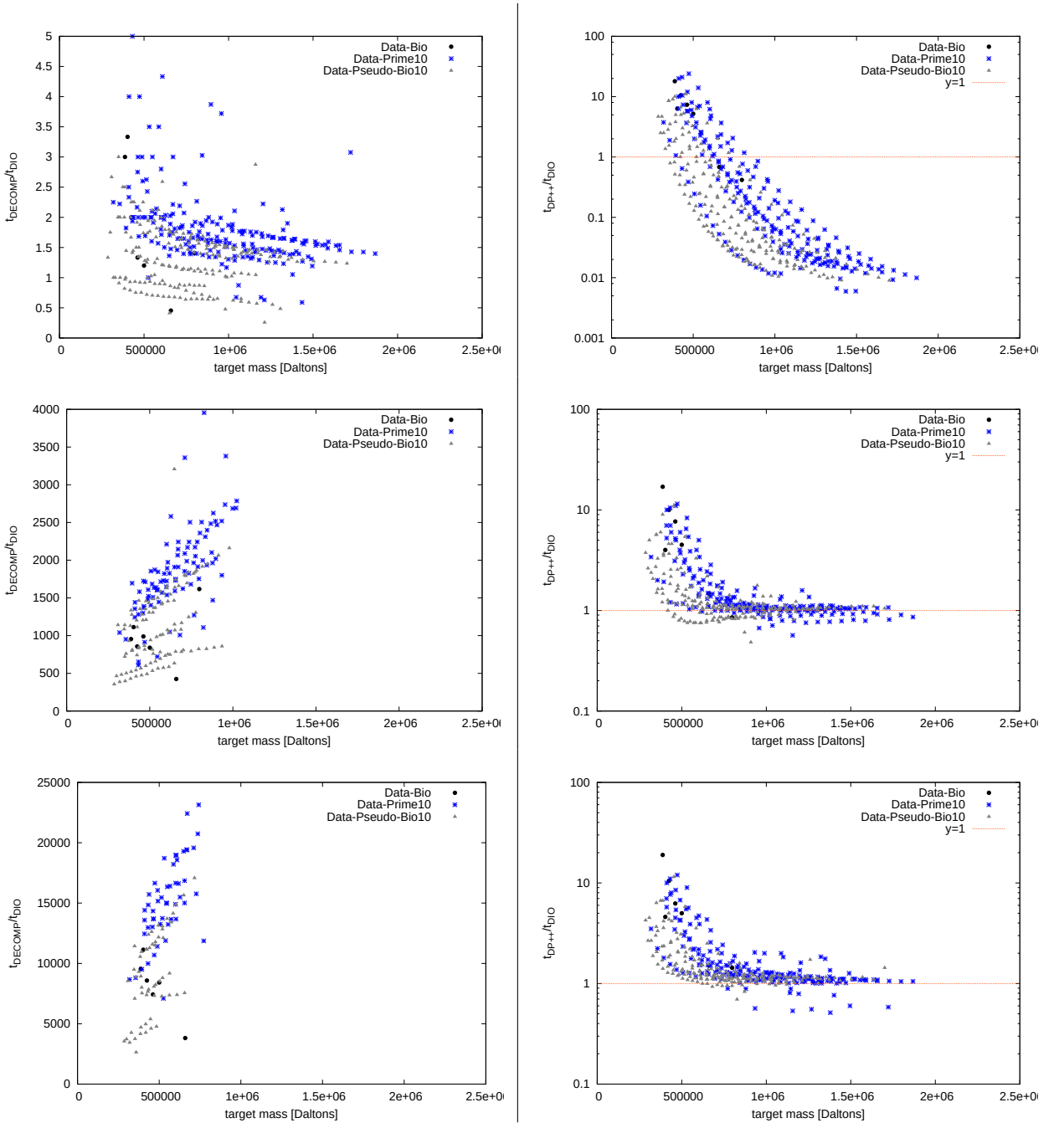


Figure 4 Number of non-negative solutions vs. asymptototic behavior of the denumerant (Eq. (4)) for Yeast Exosome. The target mass is expressed in units of the Frobenius number. Instances with a target mass beyond $11g_0$ did not terminate within the time limit. *Note that Y-axis is drawn with a logarithmic scale.*



7.4 Running times

Figure 5 DIOPHANTINE vs. DECOMP vs. DP++: running time as a function of the target mass at different noise levels The left column compares $t^{\text{Tot}}_{\texttt{DIO}}$ and $t^{\text{Tot}}_{\texttt{DECOMP}}$, while the right one compares $t^{\text{Tot}}_{\texttt{DIO}}$ and $t^{\text{Tot}}_{\texttt{DP++}}$ (y-axis:log scale). The 3 rows respectively correspond to the noise levels 0%, 0.1%, and 1%. *Note that Y-axes for the figures in the right column are drawn with a logarithmic scale.*



7.5 Output sensitivity Analysis of Algorithm DIOPHANTINE

Figure 6 DIOPHANTINE: running time $t^{\mathbf{Tot}}_{\mathtt{DIO}}$ as a function of the number of solutions. (Left) Noise level of 0.1% (Right) Noise level of 1%. *Note that Y-axes are drawn with a logarithmic scale.*

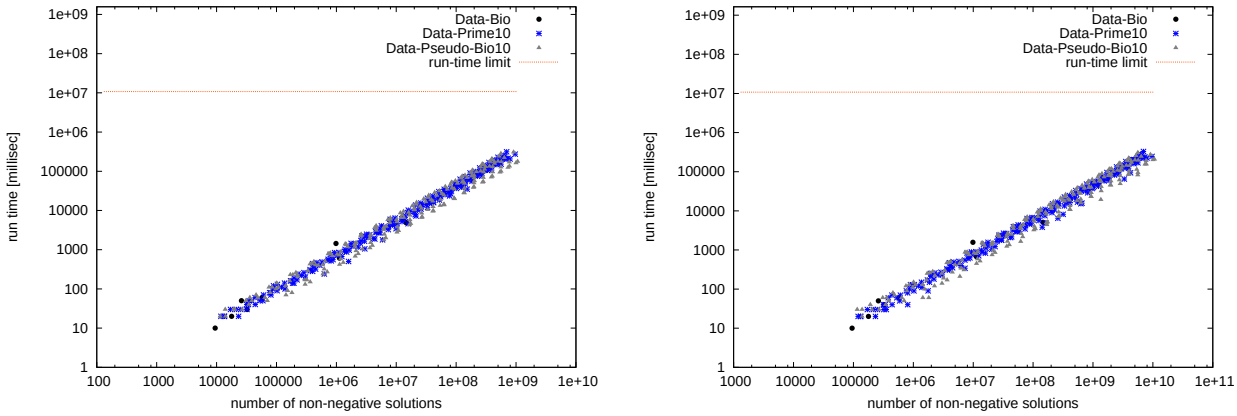


Figure 7 DIOPHANTINE: running time $t^{\mathbf{Tot}}_{\mathtt{DIO}}$ as a function of the recursion tree size. (Left) Noise level of 0.1% (Right) Noise level of 1%. *Note that Y-axes are drawn with a logarithmic scale.*

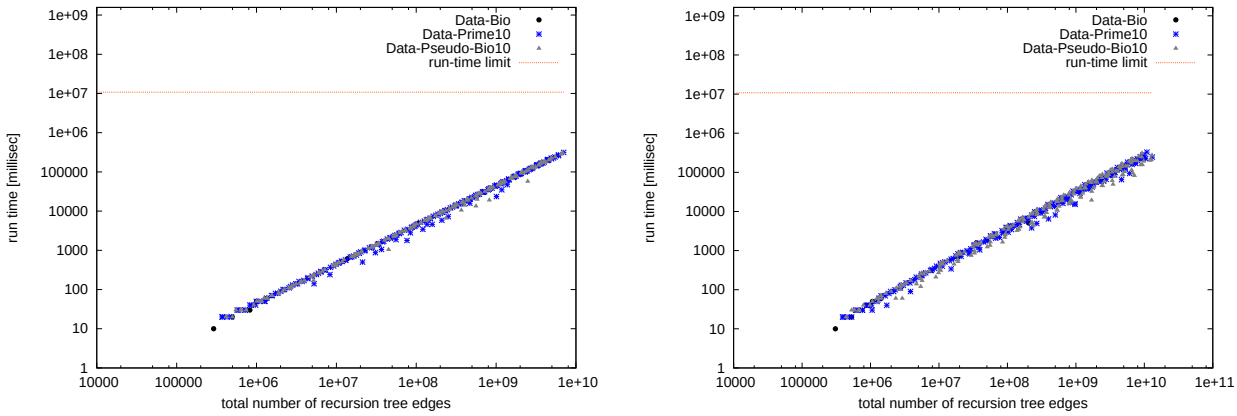
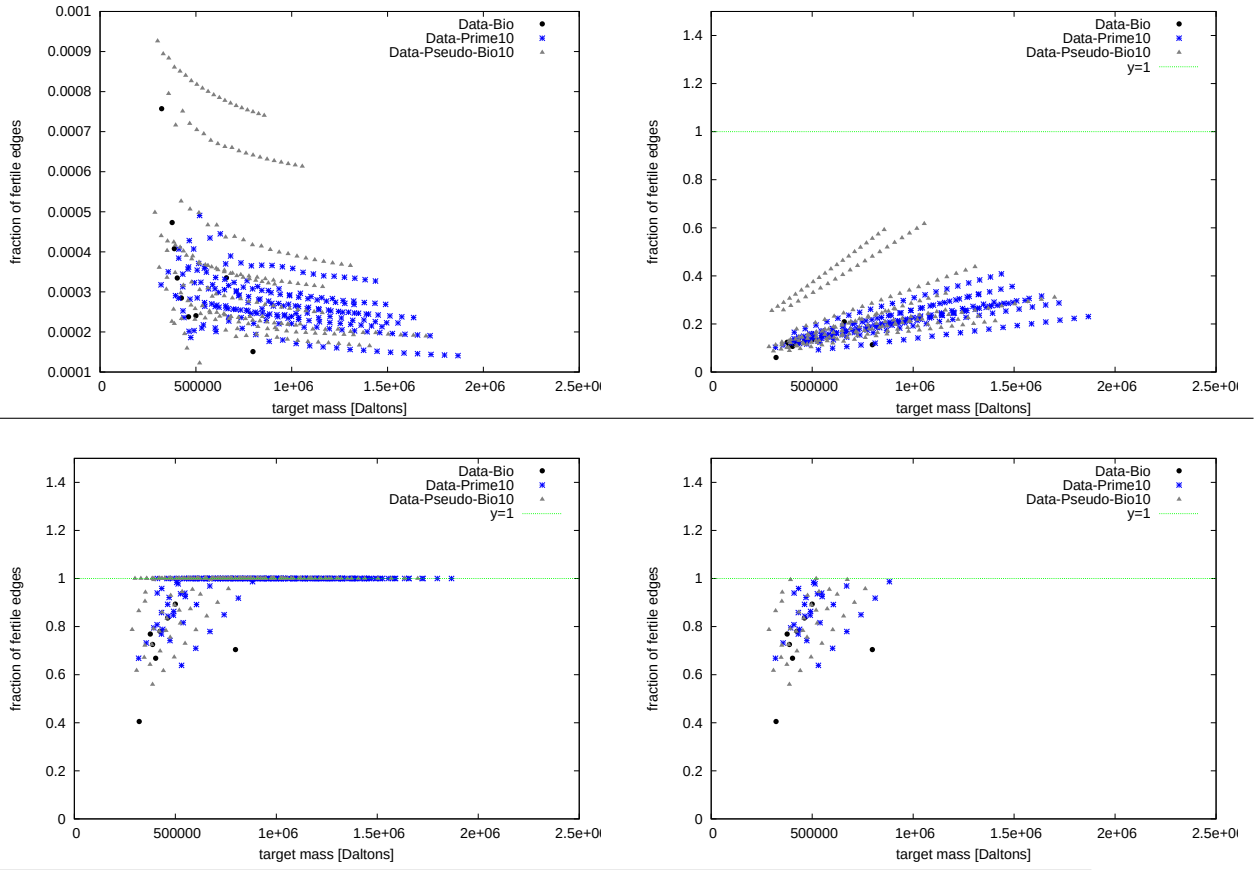


Figure 8 DIOPHANTINE**: fraction of fertile edges at three noise levels, 0% (Top-Left), 0.1% (Top-Right), 1%(Bottom-Left).** The last figure pertains to those instances with 1% noise, which do not meet with the output sensitivity criterion stated in Eq. (19).



7.6 Decreasing performance of DECOMP

Figure 9 DECOMP **[Top] and** DP++**[Bottom]: post-processing time (i.e.** $t^{\mathbf{Post}}$**) as a function of the pre-processing time (i.e.** $t^{\mathbf{Pre}}$**).** *Note that Y-axes are drawn with a logarithmic scale.*

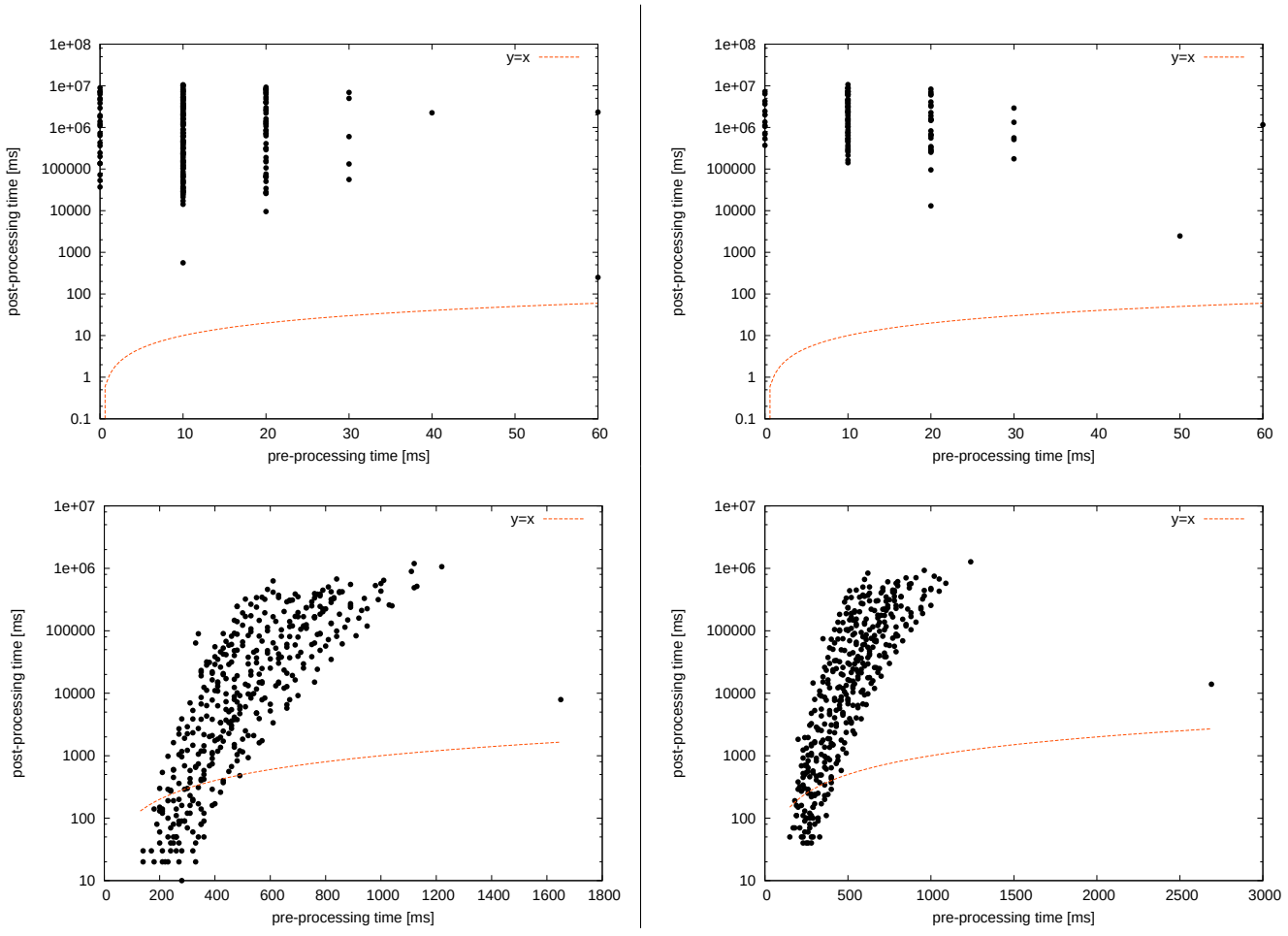


Table 3 DP++: post-processing time (i.e. $t^{\mathbf{Post}}_{\texttt{DP++}}$) dominates the pre-processing time (i.e. $t^{\mathbf{Pre}}$) for instances with large #Nodes and large #Solutions. Numbers provided are Pearson correlation coefficient at 0.1% and 1% noise level, respectively.

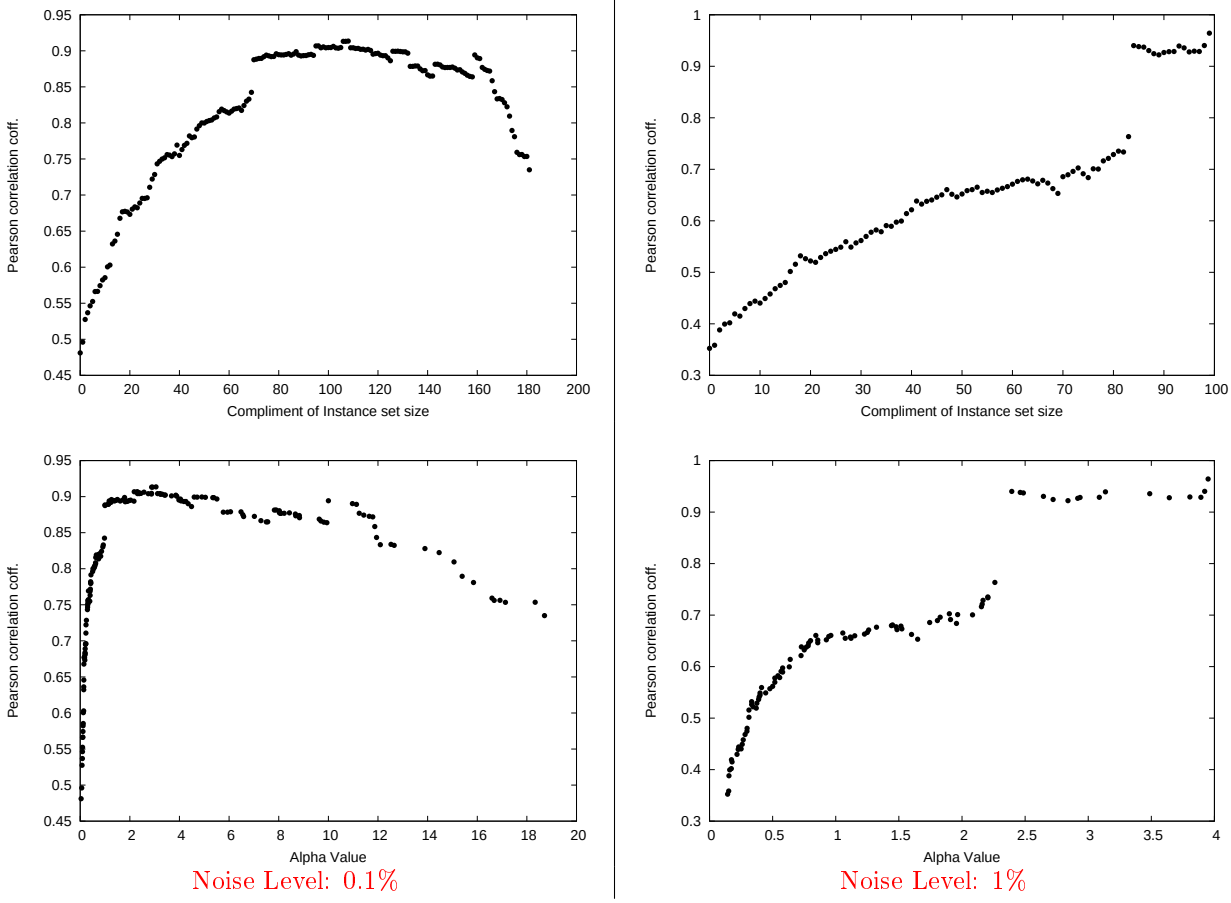
	#Nodes		*#Solutions*		*Target Mass*		$\frac{\#Solutions}{Target\ mass}$	
Noise level	0.1%	1%	0.1%	1%	0.1%	1%	0.1%	1%
$\frac{t^{\text{Post}}_{\texttt{DP++}}}{t^{\text{Pre}}_{\texttt{DP++}}}$	0.93	0.95	0.93	0.91	0.32	0.11	0.94	0.91

Table 4 DP++ and DECOMP: increase in $t^{\mathbf{Post}}_{\texttt{DECOMP}}$ over $t^{\mathbf{Post}}_{\texttt{DP++}}$ is due to relative size of trees (#nodes) for both the algorithms. Pairwise Pearson correlation coefficients to compare DECOMP and DP++ at two noise levels. (See also figs. SI-13 – SI-16).

Noise Level – 0.1%				
	$\frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}$	$\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}}$	$\frac{t^{\text{Pre}}_{\texttt{DECOMP}}}{t^{\text{Pre}}_{\texttt{DP++}}}$	$\frac{t^{\text{Post}}_{\texttt{DECOMP}}}{t^{\text{Post}}_{\texttt{DP++}}}$
error width	0.37	0.74	-0.21	0.16
$\frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}$		0.48	0.46	0.75
$\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}}$			-0.01	0.43
$\frac{t^{\text{Pre}}_{\texttt{DECOMP}}}{t^{\text{Pre}}_{\texttt{DP++}}}$				0.49

Noise Level – 1%				
	$\frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}$	$\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}}$	$\frac{t^{\text{Pre}}_{\texttt{DECOMP}}}{t^{\text{Pre}}_{\texttt{DP++}}}$	$\frac{t^{\text{Post}}_{\texttt{DECOMP}}}{t^{\text{Post}}_{\texttt{DP++}}}$
error width	0.51	0.66	-0.14	0.58
$\frac{\#nodes_{\text{DECOMP}}}{\#nodes_{\text{DP++}}}$		0.35	0.29	0.79
$\frac{t^{\text{Tot}}_{\texttt{DECOMP}}}{t^{\text{Tot}}_{\texttt{DP++}}}$			-0.13	0.73
$\frac{t^{\text{Pre}}_{\texttt{DECOMP}}}{t^{\text{Pre}}_{\texttt{DP++}}}$				0.12

Figure 10 The parameterized correlation between the ratios of total running times $t^{\mathbf{Tot}}_{\mathtt{DECOMP}}/t^{\mathbf{Tot}}_{\mathtt{DP++}}$ and $\#nodes_{\mathbf{DECOMP}}/\#nodes_{\mathbf{DP++}}$ increases upon stepwise removal of instances. (Top) At an x-coordinate $= k$, the Pearson correlation value corresponds to the $N-k$ instances of the set $S_{>i}$, see Eq. (29). **(Bottom)** The x-axis features the sorted α_i values.



8 Supporting Information

8.1 Detailed Description of Biological Complexes

Yeast 19S Proteasome lid. Proteasomes are protein assemblies involved in elimination of damaged or misfolded proteins and the degradation of short-lived regulatory proteins. They are found in all eukaryotic cells and archaea, and also in selected bacteria. The most common form of the proteasome is 26S, named after its sedimentation coefficient—expressed in Svedberg.

The 26S Proteasome, consists of one core particle corresponding to the degradation chamber (the 20S) and of two regulatory caps filtering the entry of the proteins (the 19S). The 19S subcomplex itself subdivides into two other subcomplexes, the base that binds directly to the 20S core particle and a peripheral lid. The latter is composed of 9 different protein types with a single copy for each type. It is involved in recognition of the polyubiquitin chain of the substrate and followed by it's deubiquitination. This substrate is then unfolded and translocated to the core particle for further degradation [STA$^+$06].

For yeast, the measured mass of the intact 19S Proteasome lid complex is 376,151 $\pm$ 369 Da. Summing the theoretical weights of the protein types amounts to 374,576 Da, i.e. the error in the measurement is 0.42%.

COP9 Signalosome. The COP9 Signalosome is a multifunctional complex primarily involved in ubiquitin mediated proteolysis linked to diverse cellular activities such as signal transduction, cell cycle progression and transcriptional regulation. It is composed of 8 protein types with a single copy for each type and shares remarkable homology with Yeast 19S proteasome lid complex.

An Electrospray MS experiment conducted on human COP9 signalosome reconstituted by coexpression in *E.coli* reveals the molecular weight for an intact complex to be 321,274 $\pm$ 35 Da. Summing the theoretical weights of the protein types amounts to 321,270 Da, i.e. the error in the measurement is 0.0012% [SMBE$^+$09].

Eukaryotic Translation factor EIF3. Eukaryotic translation initiation factors (EIFs) are the proteins involved in assembling of elongation competent 80S ribosome to initiate the translation process. There are atleast nine EIFs involved in the initiation process. They carry out their function in two steps: formation of 48S complex with established codon-anticodon base pairing in the P-site of 40S ribosomal subunits, and the joining of 48S complex with 60S subunits [JHP10].

Among them, EIF3 binds to the 40S subunit of the ribosome to initiate protein synthesis followed by recruitment of messenger RNAs. It also promotes attachment of 43S complexes to mRNA. It has 13 different protein types with a single copy for each type. These 13 protein types have been unambiguously identified when MS/MS spectra, and were scanned against UniProt/Swiss-Prot and NCBInr database using the MASCOT search engine [DFZ$^+$07]. The measured mass of the intact Yeast EIF3 complex is 797,999 $\pm$ 180 Da. Summing the theoretical weights of the protein types amounts to 793,558 Da, i.e. error w.r.t. to the theoretical weights is 0.56% [ZSF$^+$08].

Yeast Exosome. The exosome is a 3'- 5' exonuclease complex involved in RNA processing and degradation. In eukaryotes it is present in the cytoplasm and nucleolus and therefore reacts with different substrates in respective compartments. It is composed of 10 different protein types each with unit stoichiometry [HDT$^+$06]. The yeast exosome complex is known to contain domains

homologous to ribonucleases e.g. RNase PH and RNase II and others, e.g. S1, KH, PINc and HRDC [ACL+02].

The measured mass of the intact Yeast exosome complex is 397,860 ± 99 Da. Summing the theoretical weights of the protein types amounts to 397,881 Da, i.e. error w.r.t. to the theoretical weights is 0.005%.

Rotary ATPases. Rotary ATPases are membrane associated molecular machines involved in energy conversion by coupling ATP hydrolysis (or synthesis) with proton (or Na^+) translocation across biological membranes. There are primarily two types of ATPases, F-type and V-type complexes each having two domains F_0, F_1 and V_0, V_1. The membrane embedded domains F_0/V_0 mediate proton (or Na^+) translocation and F_1/V_1 are involved in ATP production or consumption respectively.

An electrospray mass spectrum is recorded for rotary ATPases from *E. hirae* (EhATPase) and *E. thermus* (TtATPase). Each of these complexes have nine different protein types but, the membrane embedded rotor for EhATPase is larger because each K type contains four transmembrane helices as compared to two transmembrane helices in corresponding L type in TtATPase.

MS experiment data was retrieved from [MR12]. EhATPase was undergone controlled disassembly resulting in formation of sub-complexes in gas phase and solution phase using collision induced dissociation (CID) and partial denaturation by manipulating the ionic strength, respectively. We choose four sub-complexes – 2,3,4 and 5 formed in the solution phase, and assumed that each subcomplex is composed of all the 9 types of proteins as does the intact complex. Measured molecular weights of subcomplexes and percentage error in measurement are shown in table 5.

Table 5 Measured molecular weights (in Da) and percentage error in measurement w.r.t. weighted sum of the theoretical masses of individual protein types complying with their known stoichiometries. Refer to fig. 6 of [MR12] and Table S1A and S1B of supplementary information of [ZMB+11]

Complex	Stoichiometries	Measured Mass (Da)	$\sum s_i w_i^t$	%error in measurement
EhATPase-sub-2	$A_3B_3CDE_2F_2G$	500,178 ± 294	499,131	0.21
EhATPase-sub-3	$A_3B_3DE_2F_2G$	461,674 ± 324	460,968	0.15
EhATPase-sub-4	A_3B_3DEFG	424,441 ± 148	423,813	0.15
EhATPase-sub-5	A_3B_3DG	387,356 ± 230	386,658	0.18
TtATPase	$A_3B_3CDE_2G_2FIL_{12}$	659,202 ± 131	657,979	0.19

Yeast Nuclear Pore Complex (NPC). The NPC is a protein assembly anchored in the nuclear envelope, regulating the nucleo-cytoplasmic transport. It is composed of $\sim$ 30 distinct proteins types each present in multiple copies, and is the largest protein assembly known to date in the eukaryotic cell [WR10, DH08]. It has eight-fold radial symmetry and consists of eight spokes, each with a cytoplasmic and a nuclear side.

While we are not aware of mass spectrometry experiments on the whole NPC, a number of sub-systems have been studied in detail. One of them is an heptameric sub-complex known as the Y-complex [FMPS+12, DDC12], which participates to the formation of the scaffold of the NPC, as one finds one copy of the Y-complex per half-spoke.

In the sequel, we simulated mass spectrometry data for two complexes. The first one is a complete spoke, which contains a total of 57 proteins instances of 30 different types. The second

one is the Y-ring complex, containing 8 copies of the Y-complex—that is 56 proteins of 7 types. For each complex, noise levels of 0%, 0.1% and 1% were applied to the exact mass of the complex computed from the masses of the individual proteins.

8.2 Plots Corresponding to the Table 3

Figure 11 DP++**: The Dominance of $t^{\mathbf{Post}}_{\mathtt{DP++}}$ over $t^{\mathbf{Pre}}_{\mathtt{DP++}}$ increases for instances having large tree size (#nodes) and large #solutions. [Top]** $t^{\mathrm{Post}}_{\mathtt{DP++}}/t^{\mathrm{Pre}}_{\mathtt{DP++}}$ as a function of number of nodes and **[Bottom]** Number of solutions, at 0.1% and 1% noise level for three datasets.

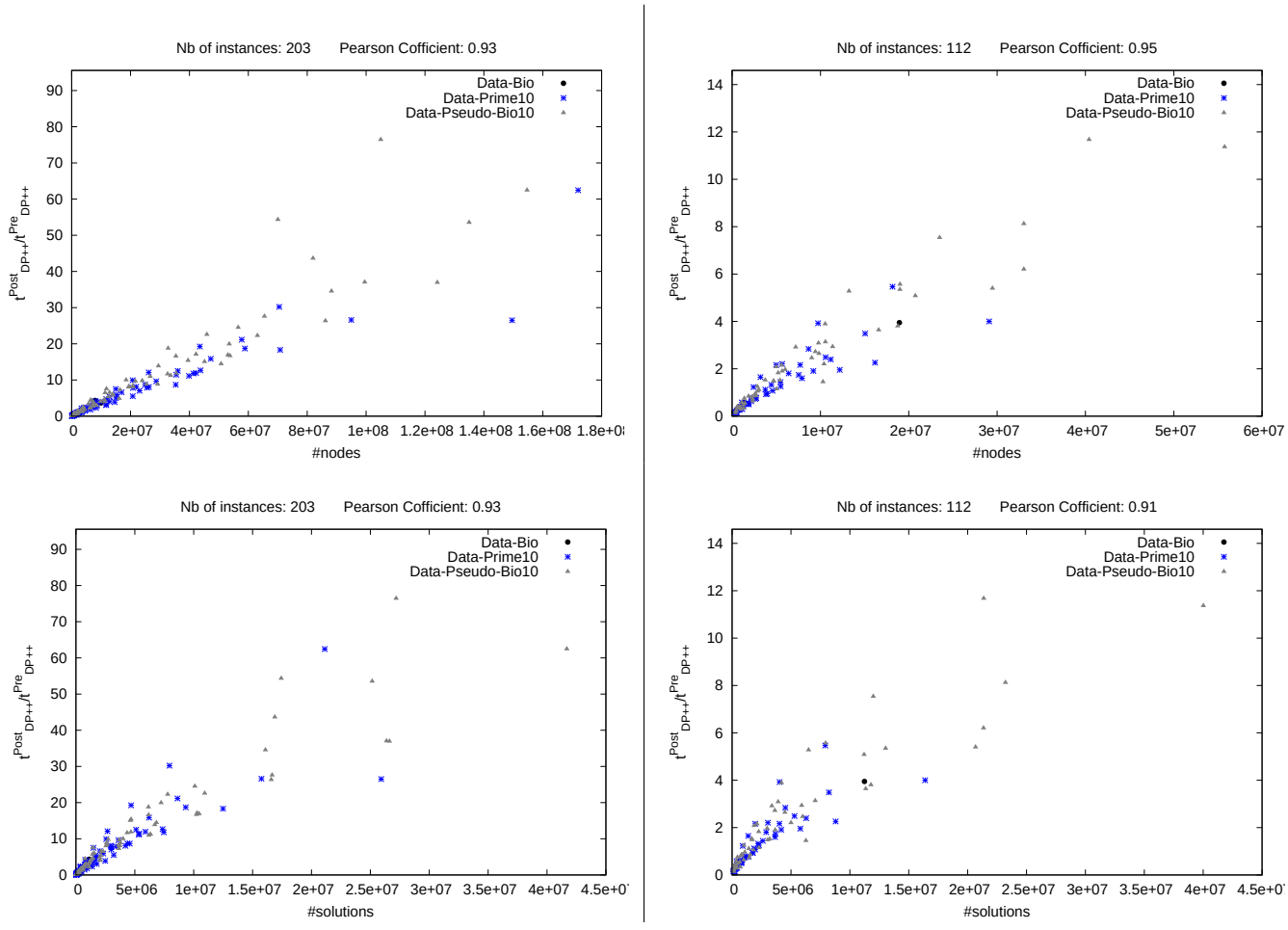
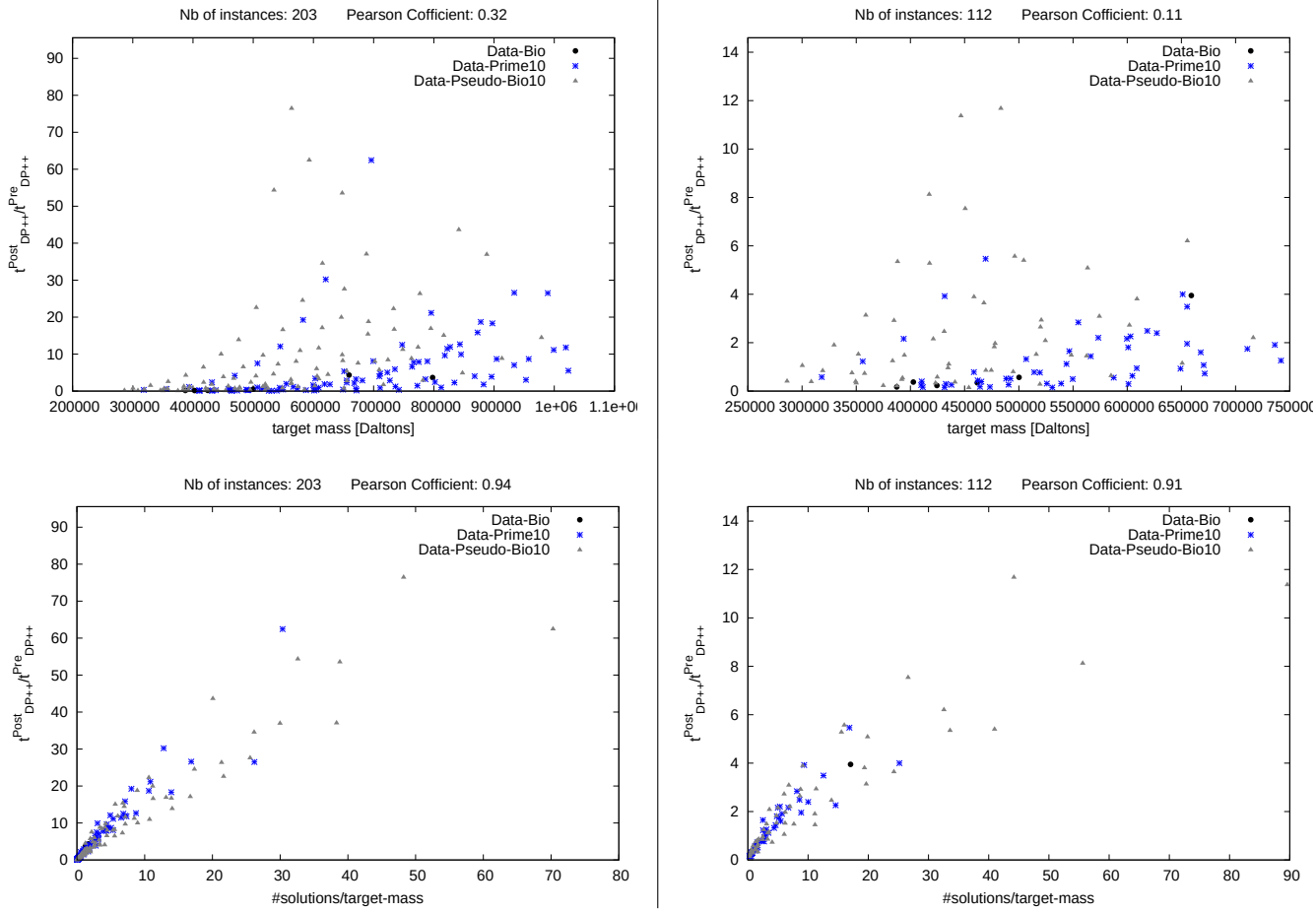


Figure 12 DP++: **Increase in target mass does not contribute in dominance of $t^{\mathbf{Post}}_{\mathtt{DP++}}$ over $t^{\mathbf{Pre}}_{\mathtt{DP++}}$. [Top]** $t^{\mathrm{Post}}_{\mathtt{DP++}}/t^{\mathrm{Pre}}_{\mathtt{DP++}}$ as a function of Target mass in Daltons and **[Bottom]** ratio of Number of solutions over Target mass, at 0.1% and 1% noise level for three datasets.



8.3 Plots Corresponding to the Table 4

Figure 13 Pairwise plots between error width and ratio of run times for DECOMP over DP++. (Columns) The left and right columns respectively correspond to noise levels 0.1% and 1%. **(Top row)** Error width vs. Ratio of total run time for DECOMP over DP++ **(Middle row)** Error width vs. Ratio of post-processing time (backtracking) for DECOMP over DP++ **(Bottom row)** Error width vs. Ratio of pre-processing time for DECOMP over DP++

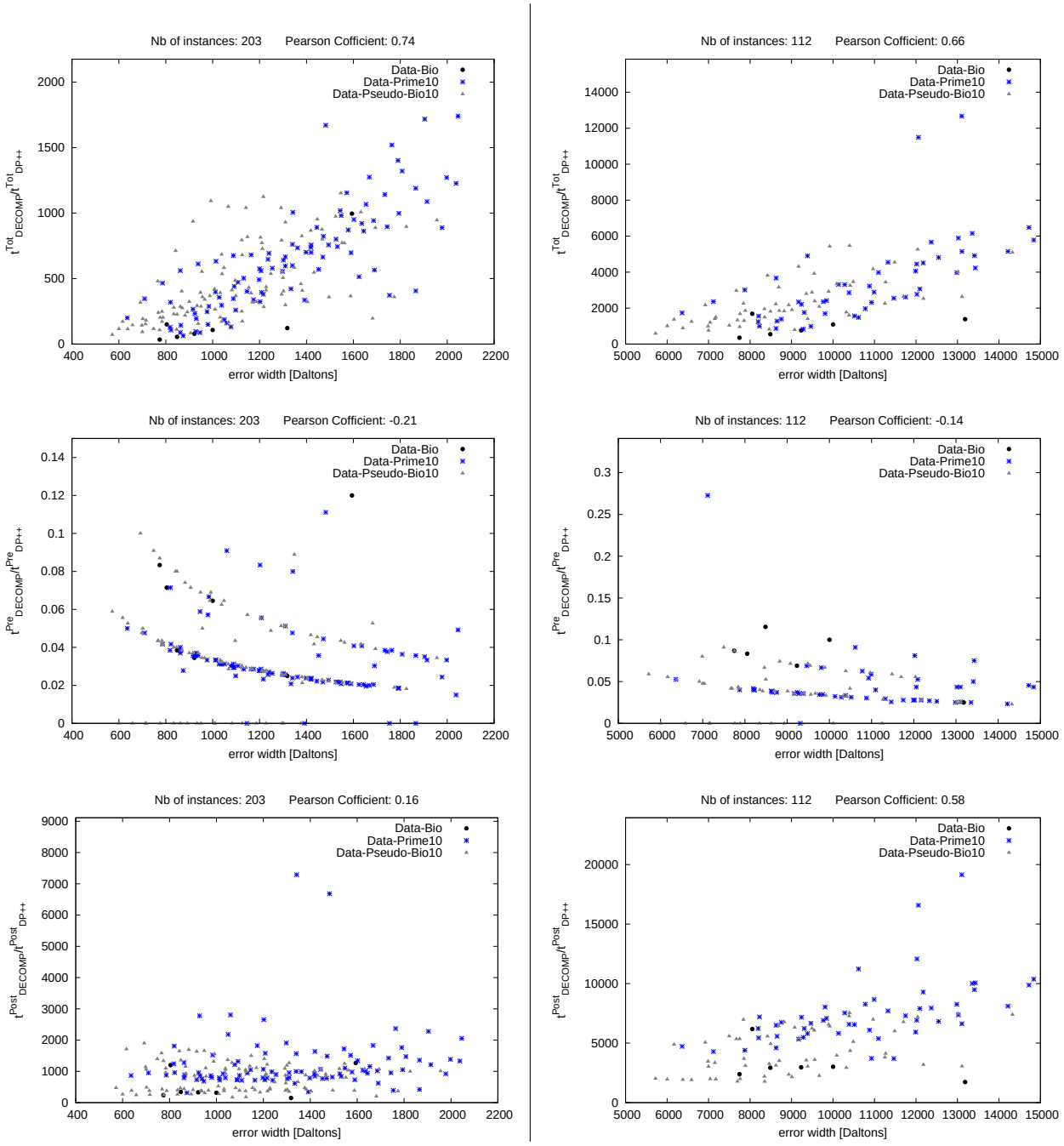


Figure 14 Pairwise plots among ratio of runtimes and ratio of number of nodes for DECOMP over DP++ (Columns) The left and right columns respectively correspond to noise levels 0.1% and 1%. **(Top row)** Ratio of number of nodes explored vs. Ratio of total run time for DECOMP over DP++ **(Middle row)** Ratio of number of nodes explored vs. Ratio of post-processing time (backtracking) for DECOMP over DP++ **(Bottom row)** Ratio of number of nodes explored vs. Ratio of pre-processing time for DECOMP over DP++

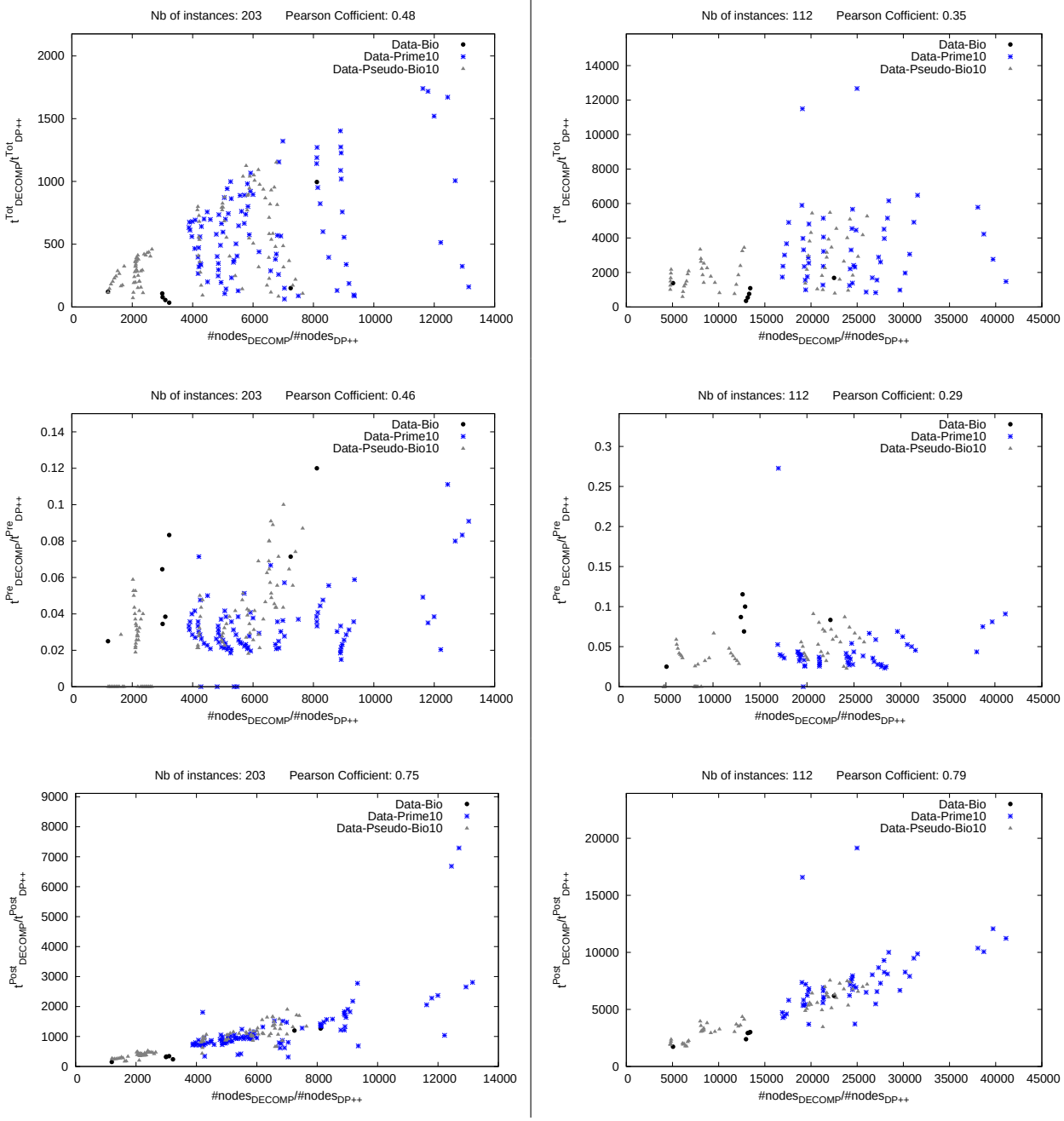
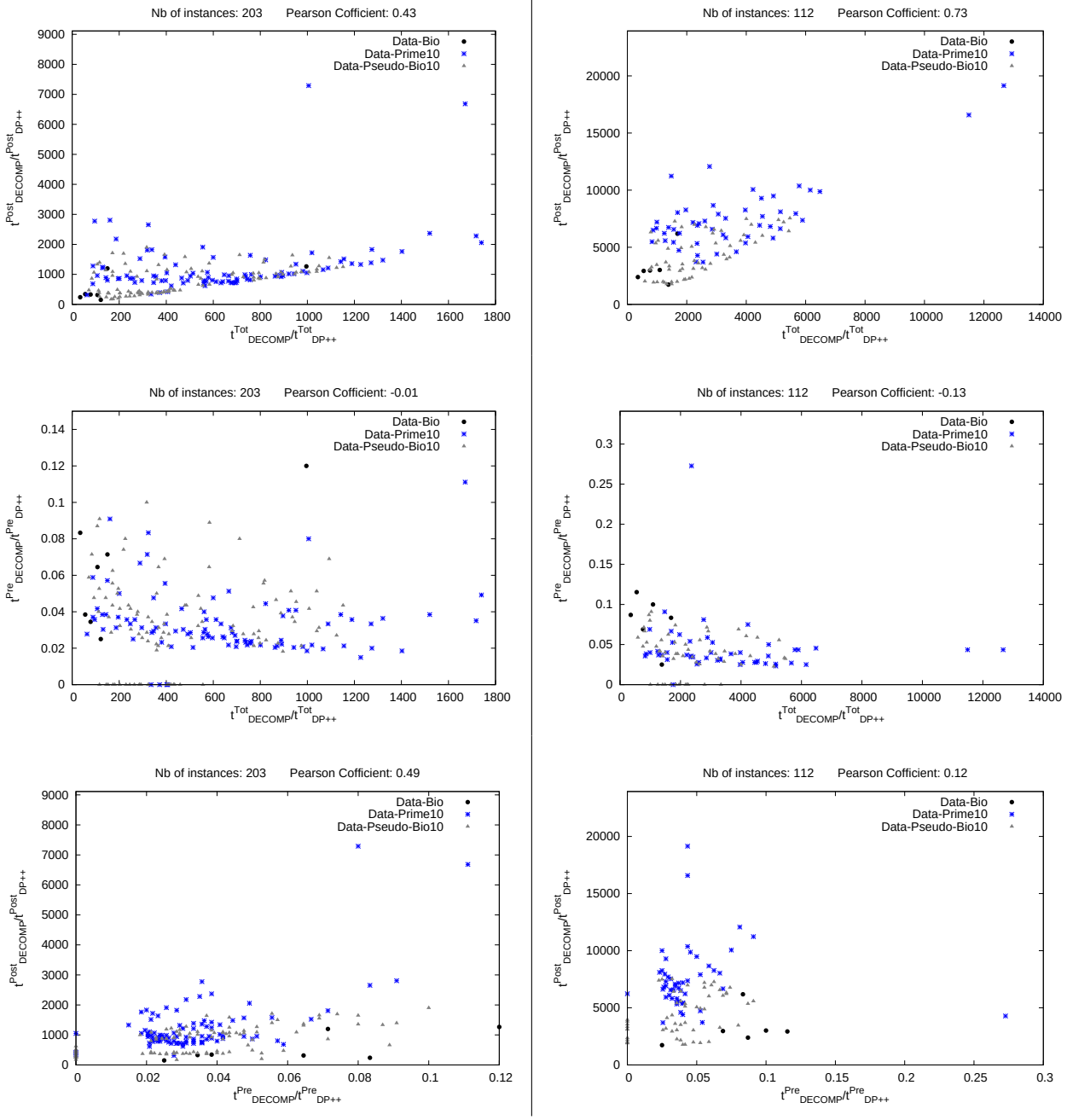
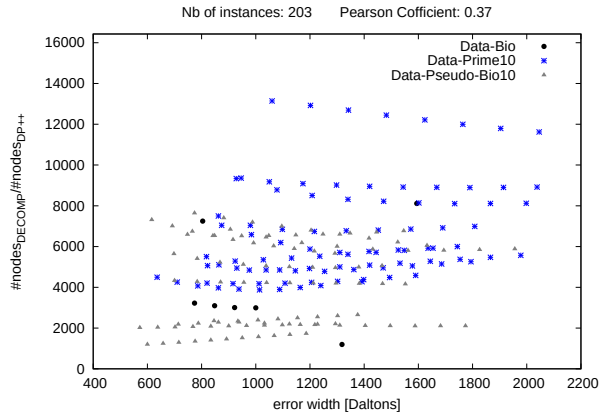


Figure 15 Pairwise plots among ratio of total run times, post-processing time and pre-processing time for `DECOMP` over `DP++`. (Columns) The left and right columns respectively correspond to noise levels 0.1% and 1%. **(Top row)** Ratio of total run time vs. Ratio of post-processing time (backtracking) `DECOMP` over `DP++` **(Middle row)** Ratio of total run time vs. Ratio of pre-processing time for `DECOMP` over `DP++` **(Bottom row)** Ratio of post-processing time (backtracking) vs. Ratio of post-processing time for `DECOMP` over `DP++`

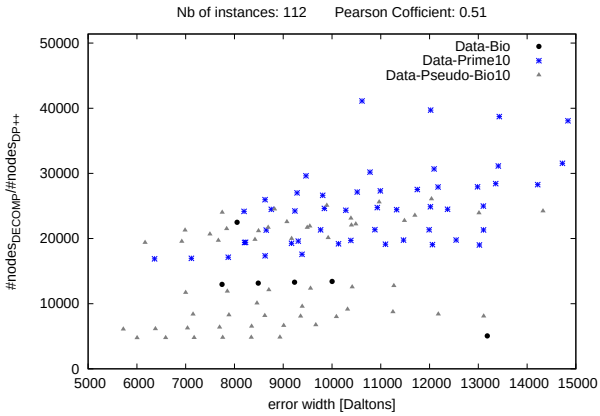


Noise Level: 0.1% | Noise Level: 1%

Figure 16 Error width vs. Ratio of number of nodes explored for DECOMP over DP++.



Noise Level: 0.1%



Noise Level: 1%

8.4 Studying the hierarchical tree for Biological complexes

Figure 17 Number of tuples of different cardinality used in the Biological complexes are close to the respective Binomial coefficients even at small noise level. Cardinality or size of the tuple corresponds to the non-negative solutions of 0.1%.

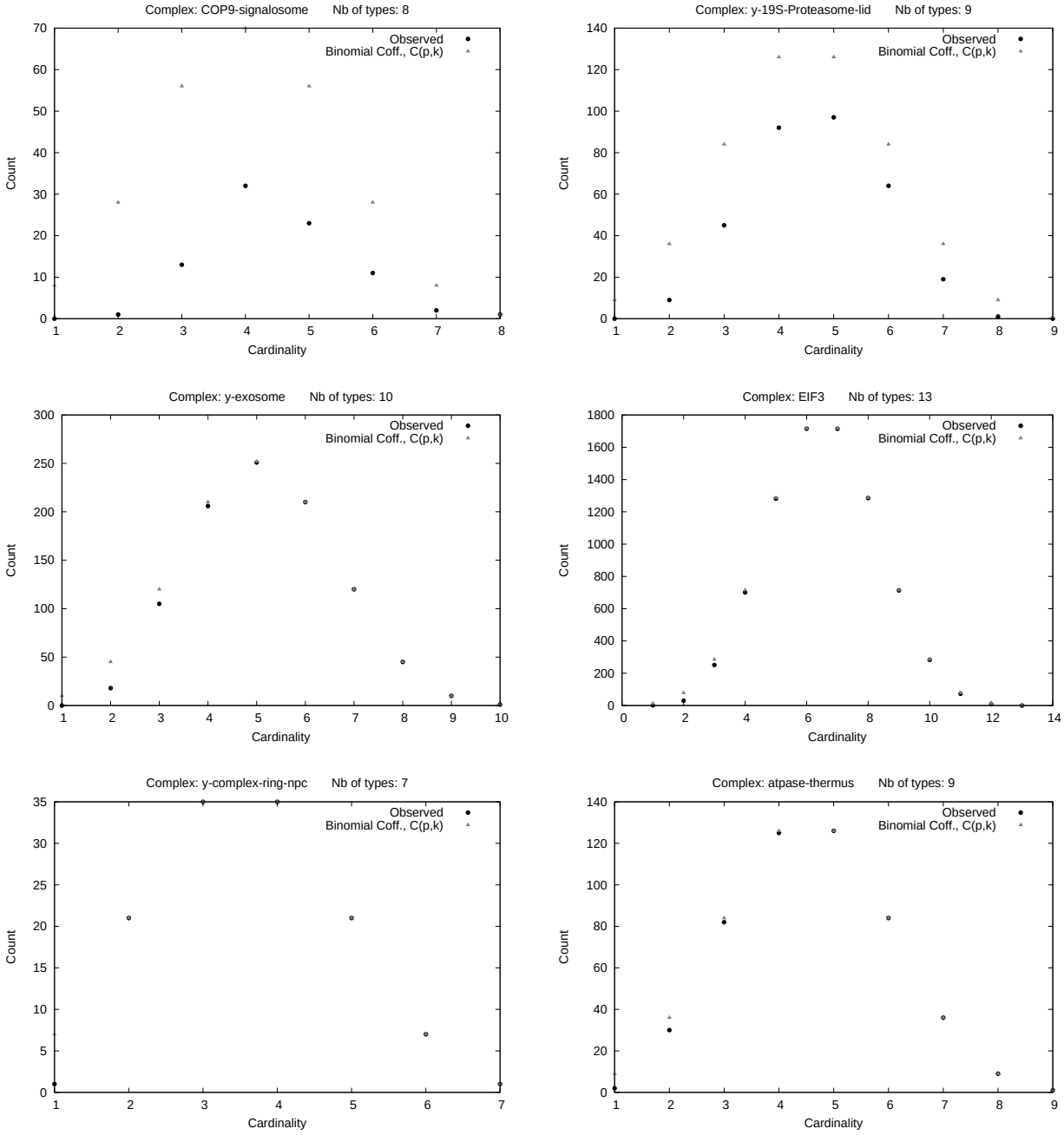
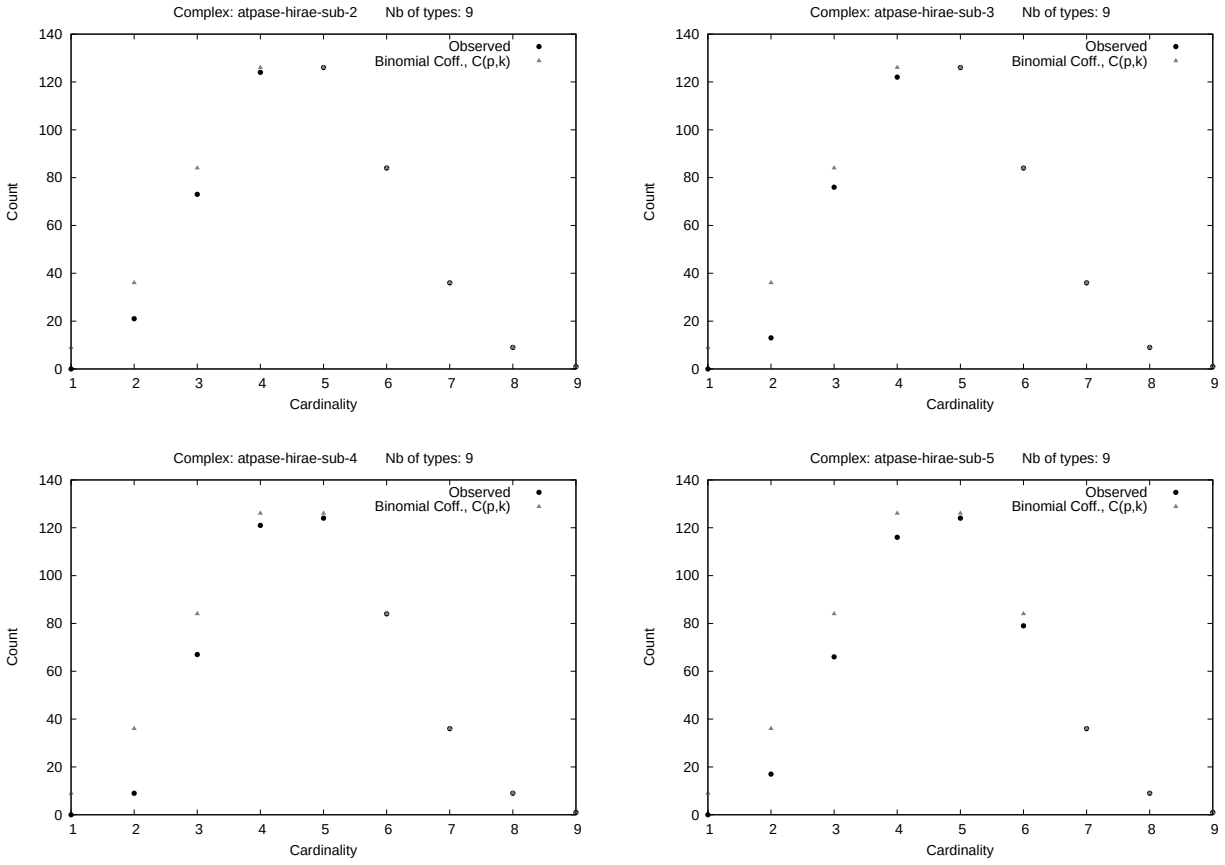
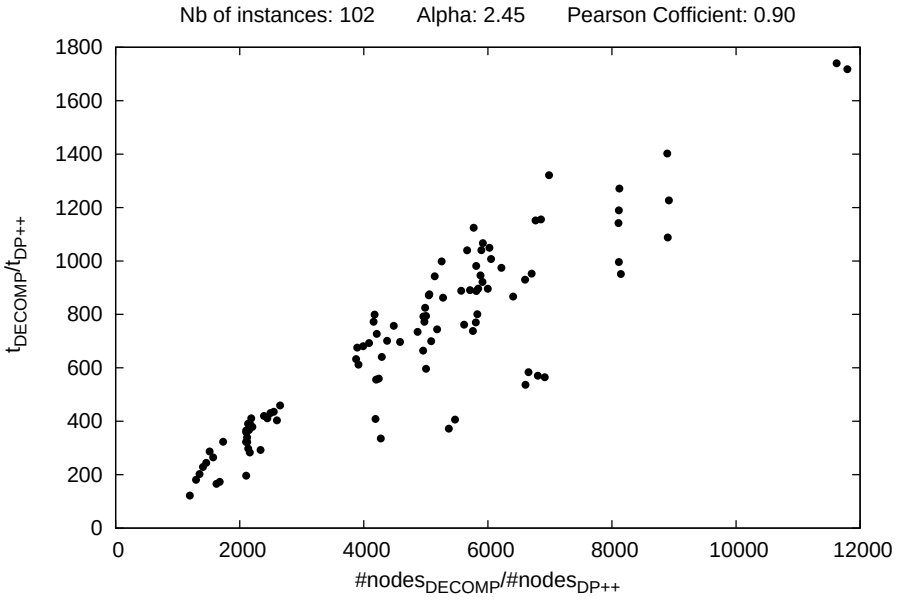


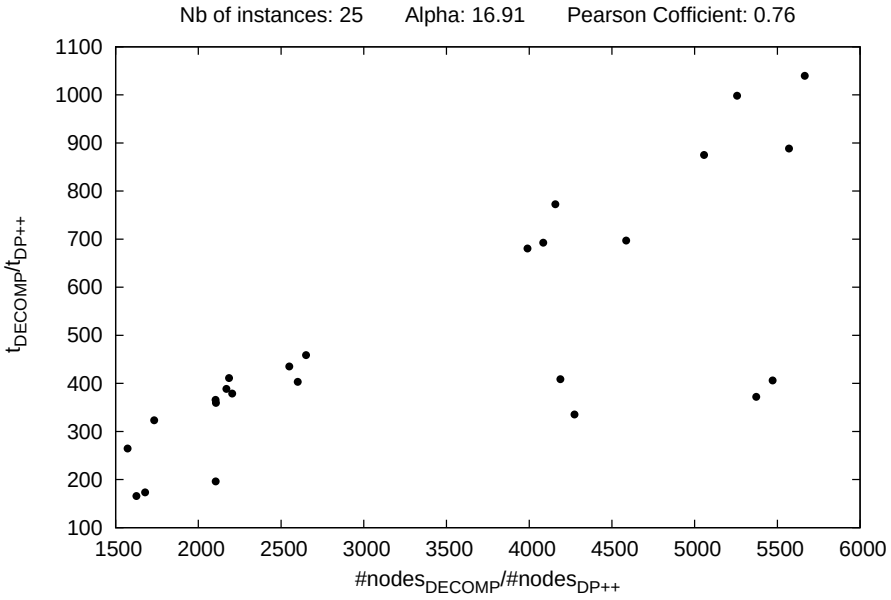
Figure 18 Number of tuples of different sizes used in the Biological complexes are close to the respective Binomial coefficients even at small noise level. Cardinality or size of the tuple corresponds to the non-negative solutions at noise level of 0.1%.



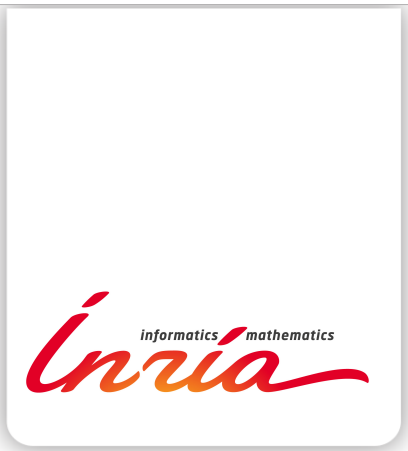
8.5 Scatter plots to compare DP++ and DECOMP

Figure 19 Paucity of data effects the Pearson Correlation Coefficient. Scatter plots corresponding to two Pearson coefficients in the Fig. 10 at 0.1% noise level. A small number of instances (bottom figure) yields a scatter plot loosely distributed, resulting in a decrease of the Pearson correlation coefficient despite the increase in alpha value.





Contents



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399