DNA computing of solutions to knapsack problems

Henkel, C.V.; Bäck, T.H.W.; Kok, J.N.; Rozenberg, G.; Spaink, H.P.

Available online at www.sciencedirect.com
ScienceDirect
BioSystems 88 (2007) 156–162
BioSystems
www.elsevier.com/locate/biosystems

DNA computing of solutions to knapsack problems

Christiaan V. Henkel [a,b], Thomas Bäck [b], Joost N. Kok [b], Grzegorz Rozenberg [b], Herman P. Spaink [a,*]

[a] *Institute of Biology, Leiden University, Wassenaarseweg 64, 2333 AL Leiden, The Netherlands*
[b] *Leiden Institute of Advanced Computer Science, Leiden University, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands*

Received 23 March 2005; received in revised form 9 May 2006; accepted 5 June 2006

Abstract

One line of DNA computing research focuses on parallel search algorithms, which can be used to solve many optimization problems. DNA in solution can provide an enormous molecular library, which can be searched by molecular biological techniques. We have implemented such a parallel search for solutions to knapsack problems, which ask for the best way to pack a knapsack of limited volume. Several instances of knapsack problems were solved using DNA. We demonstrate how the computations can be extended by *in vivo* translation of the DNA library into protein. This combination of DNA and protein allows for multi-criterion optimization. The knapsack computations performed can then be seen as protein optimizations, one of the most complex computations performed by natural systems.

Keywords: DNA computing; Aqueous computing; Knapsack problem

1. Introduction

Plasmids (naturally occurring circular DNA molecules) can serve to perform computations at the molecular level (see Paun et al., 2005, for an introduction to molecular computing). Specially designed plasmids contain a dedicated computing region, which is basically a computer memory with bits set to either 1 or 0. A molecular memory consisting of many plasmids in solution can be operated on by restriction endonucleases. Removal of a region from the plasmid (a 'station') is identified with the flipping of a bit (Head, 2000). In theory, all plasmids in a solution can represent different memory configurations and therefore provide a huge parallel search space for optimization problems.

Typically, a computation starts with a single species of plasmid, from which a library of different plasmids is generated by repeated modifications to subsets of the plasmid mixture. From this library, a memory configuration corresponding to the solution to a certain problem can be selected using molecular biological separation technologies. Plasmid computing has been successfully applied to small test instances of several computationally hard optimization problems, and can be applied to a broad range of algorithmic problems (Head et al., 2000, 2002).

Recently, we extended plasmid computing with protein expression by the construction of the whole computing region of a plasmid as part of an open reading frame (Henkel et al., 2005). After library generation, the library was expressed into a protein representation, and this was in turn used to select a solution. A potential advantage of this translation of the solution into protein are smaller molecules (which can be analyzed using sen-

* Corresponding author. Tel.: +31 71 5275055.
E-mail address: spaink@rulbim.leidenuniv.nl (H.P. Spaink).

0303-2647/$ – see front matter © 2006 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.biosystems.2006.06.001

sitive mass spectrometry technology) and consequently higher information densities.

In this computation both DNA and protein encoded exactly the same information, i.e. the values of bits. However, the translation process can also be taken advantage of by specifying different information in DNA and protein. In this way, the optimization of the solution may be realized according to multiple distinct criteria. On the nucleic acid level, DNA length, presence and sequence are obvious encoding possibilities. Amino acids allow for linear representation of these after translation, but add more complex physico-chemical characteristics such a molecular weight and isoelectric properties.

Here, we have solved several instances of knapsack problems using plasmids. These problems ask for the best way to pack a knapsack of limited capacity with items of different size and weight. DNA computing seems very well suited for this family of problems, as both the encoding and the algorithm are relatively straightforward: formal size or weight can be linked directly to physical properties of DNA. In addition, because these problems allow for multi-criterion optimization, both DNA length and protein mass can be used to compute.

2. Results

2.1. Knapsack problems

The knapsack family of problems asks for ways to pack a volume (knapsack) of limited capacity in the most efficient way. The solution depends on the nature of the items to be packed. In some variants, only size is associated with the items—in others, items have both sizes and values, and the efficiency is evaluated by the total value packed in the fixed size knapsack.

The problem can be defined as follows: given a set of n items, each with size $s_i \in N$ (natural numbers including 0) and value $v_i \in N$, and a knapsack capacity C, maximize:

$$\sum_{i=1}^{n} x_i v_i \quad \text{subject to} \quad \sum_{i=1}^{n} x_i s_i \leq C,$$

where x_i is the multiplicity of item i in the knapsack. In the binary or 0/1 knapsack problem treated here, the availability of each item is limited to one, so $x_i \in \{0,1\}$. In the unbounded or integer knapsack problem, the supply of each item is unlimited: $x_i \in N$. If only item size is considered ($s_i = v_i$ for every item i), the binary knapsack problem reduces to the subset sum problem:

$$\sum_{i=1}^{n} x_i s_i \leq C, \quad x_i \in \{0, 1\}.$$

All these problems belong to the class NP-complete, which means that no deterministic algorithm is known to solve them in polynomial time, and it is unlikely such an algorithm exists (Garey and Johnson, 1979). However, algorithms exist that can solve certain knapsack variants in pseudo-polynomial time, i.e. in practice their exponential scaling behaviour can be rather mild. Still, the only guaranteed way to solve instances of the knapsack problem is the exhaustive enumeration of all possible knapsack packings.

2.2. Algorithm

To implement knapsack problems using plasmids, stations in the computational plasmid (Fig. 1) are identified with items in the knapsack. Other than in previous computations using a similar plasmid (see for instance Henkel et al., 2005), presence or absence of the stations is not associated with bit values. Instead, knapsack item sizes are encoded by the length of these stations (the number of basepairs). The items available in plasmid pKnapsack1 are listed in Table 1.

The experimental algorithm (Fig. 2a) starts with an initial (overstuffed) plasmid knapsack containing all

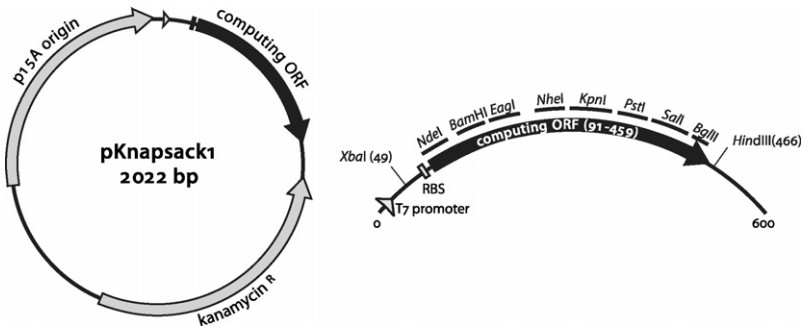


Fig. 1. Plasmid pKnapsack1. The computing ORF contains eight stations, each of which can be excised by the indicated enzyme.

Table 1
pKnapsack1 elements

Item	Plasmid fragment (bp)	Length (bp)	Weight (kDa)	Excision enzyme	Epitope
1	424–445	21	0.55	*Bgl*II	7× Gly
2	373–418	45	1.74	*Sal*I	VSV-G
3	331–367	36	1.32	*Pst*I	Thrombin/enterokinase
4	274–325	51	1.89	*Kpn*I	S-tag
5	232–268	36	1.44	*Nhe*I	FLAG
6	178–214	36	1.36	*Eag*I	c-myc
7	88–124	36	1.43	*Nde*I	HA
8	136–172	36	1.41	*Bam*HI	8× His

eight items. A library of all possible knapsack packings was generated by the iterated removal of stations from subsets of the plasmid mixture (see Fig. 2b). The station flanked by *Bam*HI sites was not used in the computation. After seven rounds of selective removal, a mixture of 128 different plasmid species was obtained (Fig. 3a). The plasmid library was then separated according to length by gel electrophoresis, and a knapsack capacity was imposed by excision of a band of the desired size. The most time-consuming step of this algorithm, the library generation, takes $O(n)$ steps for a knapsack with n available items. This is an exponential speedup from the sequential generation of all possibilities, which takes $O(2^n)$ steps.

2.3. Subset sum computation

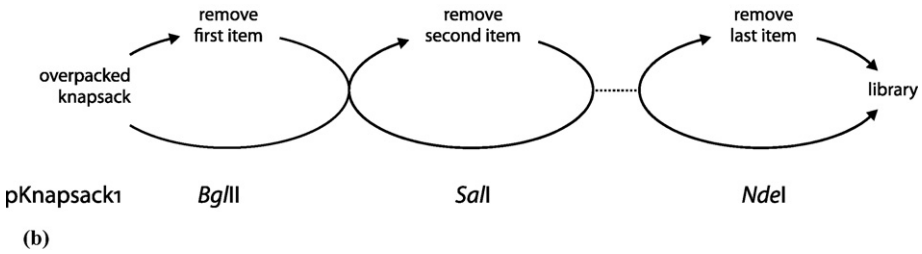
The subset sum problem considers only item size, and items can occur only once. Therefore, a subset can be obtained by excision of a band corresponding to a certain sum from the separated library. For this computation, *Xba*I/*Hin*dIII fragments of 260 and 330 bp were selected.

```
knapsack initialization:
for every item
    set item multiplicity to 1 (present)
endfor

knapsack library generation:
for every item
    divide the knapsack library into two sublibraries
    in one sublibrary, set item multiplicity to 0 (absent)
    join the two sublibraries
endfor

knapsack selection:
separate knapsack library by size
if there exists a knapsack of capacity C then
    for every item in that knapsack
        output item multiplicity
    endfor
    else output 'no solution'
endif
```

(a)



(b)

Fig. 2. Generation of all possible knapsack fillings by digestion and ligation. (a) Pseudo-code algorithm for the subset sum computation. (b) Schematic representation of the library generation phase. For every item, half of the mixture of plasmids is left untreated. In the other half, the station is removed. After this procedure, the two solutions are mixed again and the next item is processed. The separation in two subsets is assumed to yield identical plasmid mixtures.

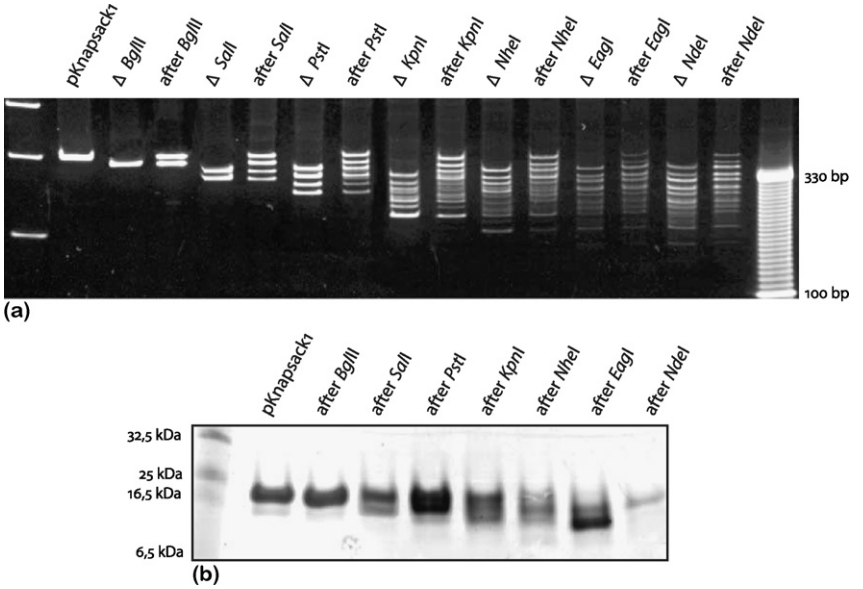


Fig. 3. The complete library of potential solutions. (a) Plasmid library. 0.5 μg plasmid was analyzed for every station removal and for every subset rejoining. The last lane (after *Nde*I) represents the final library. DNA size markers: Smart Ladder (Eurogentec) and 10 bp ladder (Invitrogen). *Xba*I/*Hin*dIII digests of the plasmids were analyzed on 4–12% TBE polyacrylamide gels (Novex precast, Invitrogen) and stained by SYBR Gold (Molecular Probes). Images were captured on a Biorad FluorS Imager using UV excitation and a 530 nm bandpass filter. The figure is a combination of two gel images. (b) Protein library. Protein was purified for every subset joining. Ten microliters of the final elution fraction was analyzed on a 10% Tris–tricine polyacrylamide gel. The gel was Coomassie stained and scanned on a Biorad GS-800 densitometer. Molecular weight marker: prestained broad range (New England Biolabs).

Because these fragments contain 156 bp of additional DNA (see Fig. 1), this corresponds to knapsack capacities of 104 and 174, respectively. The fragments were isolated from gel, religated into the vector, and introduced into *E. coli*. Individual plasmid species were physically separated by plating of the transformants.

For capacity 104, plasmids from five colonies were reisolated (Fig. 4a) and analyzed using restriction enzymes. The first lane contains a plasmid with a 264 bp insert, containing the *Nde*I, *Nhe*I and *Pst*I items. This corresponds to a subset sum capacity of 108. Two plasmids (lanes 2 and 3) were found to contain the *Nde*I, *Sal*I and *Bgl*II stations, summing up to 258 bp (or a subset sum capacity of 102). Two other plasmids (lanes 4 and 5) contain the *Sal*I and *Kpn*I stations, summing up to 252 bp. A fragment of exactly 260 bp was not recovered, and exhaustive (non-molecular) enumeration of possibilities indicates that it does not exist.

Capacity 174 was analyzed in the same way (Fig. 4b). Recovered plasmids contained a 345 bp insert (lanes 1 and 2), composed of the *Nde*I, *Kpn*I, *Pst*I, *Sal*I and *Bgl*II stations or the *Nde*I, *Nhe*I, *Kpn*I, *Sal*I and *Bgl*II stations, respectively. Lanes 3–5 show 324 bp inserts, composed of either the *Nde*I, *Nhe*I, *Kpn*I and *Sal*I stations (lane 3) or the *Eag*I, *Nhe*I, *Kpn*I and *Sal*I stations. No fragment of exactly 330 bp was recovered, although it should exist (containing the *Bgl*II, *Sal*I and three 36 bp stations).

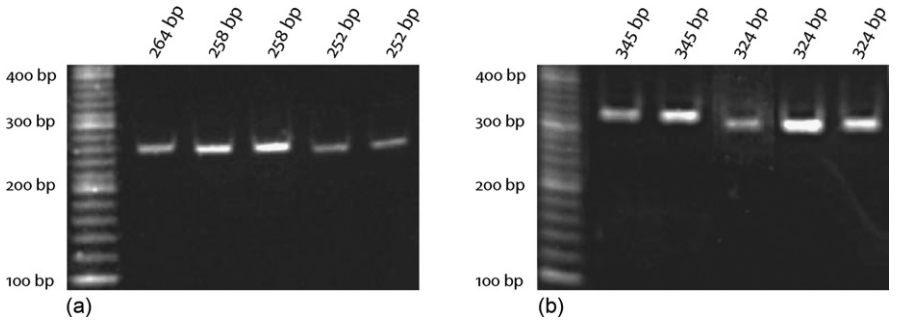


Fig. 4. Subset sum computation. *Xba*I/*Hin*dIII digests of five different plasmids with a 260 bp (a) or 330 bp (b) knapsack capacity imposed, separated on a 4–12% gel and imaged as before. DNA size marker: 20 bp ladder (Sigma).

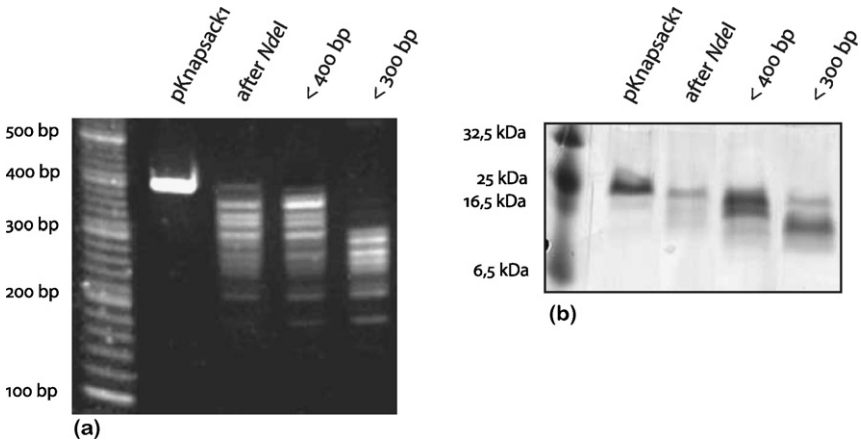


Fig. 5. Binary knapsack computation. (a) Size limits: from the original library (after *Nde*I), two pools were generated with maximum sizes of 400 and 300 bp, respectively. The largest fragment in the complete library is the original pKnapsack fragment, 417 bp. *Xba*I/*Hin*dIII fragments, imaged as before. (b) Values: translation products from the plasmid pools of (a). Silver stained 10% Tris–tricine polyacrylamide gel, scanned as before.

2.4. Binary knapsack computation

Because the computational region of the library plasmids is structured as a single ORF under the control of a strong inducible promoter, the computation can be extended to include protein optimization. The molecular binary knapsack problem considered here asks for the maximum protein mass that can be realized for a certain maximum DNA length (analogous to a maximum total value for a certain size capacity). The DNA knapsack capacity was again limited by band excision, however, in this case also knapsack fillings below the maximum capacity were considered. In the presence of heterogeneous items, maximum protein value is not guaranteed by maximum DNA size.

Knapsack capacities were chosen at 144 and 244, corresponding to fragments up to 300 and 400 bp, respectively (Fig. 5a). The mixture of fragments was again religated into the vector and introduced into *E. coli*. The resulting bacterial culture was used for protein expression. Protein was purified from the culture using the affinity of the eight histidine residues present in all computational proteins (encoded by the *Bam*HI station, see Table 1) for nickel ions. Fig. 3b shows the translated and purified product of the different steps during library construction. Fig. 5b shows the translation products of the different knapsacks selected.

The translation product of the pKnapsack1 computing ORF has a predicted mass of 13.5 kDa, a completely empty knapsack yields a protein of less than 4 kDa. Protein gel electrophoresis does not provide the accuracy to characterize such molecules. Figs. 3b and 5b can only be used to establish a correlation between knapsack size and value. The identity of the heaviest protein may be determined by mass spectrometry methods, or using antibodies specific for the different epitopes.

3. Discussion

We have demonstrated how DNA computing can be used to solve small-scale instances of knapsack problems. So far, this family of problems has been largely overlooked in the biomolecular computing field. They have received some theoretical attention, but always in relation to models of computation that have not yet been physically implemented (Chang et al., 2004; Pérez-Jiménez and Sancho-Caparrini, 2002). Two three-item subset sum experiments have been reported, but both approaches appear to have met their practical limits at this size (Aoi et al., 1998; Stoschek et al., 2001). Knapsack problems allow for the most natural encoding in DNA of all molecular computations reported so far. There is no need for elaborate sequence design as seen in models relying on hybridization, and even the mapping of sequence to formal logic can be dispensed with: the physical dimensions of the DNA used immediately suggest the problem ingredients. Also, the knapsack algorithm makes unprecedented use of DNA's potential parallelism. By separating the plasmid library according to length, many simultaneous knapsack computations take place. In fact, both subset sum capacities enforced in the experiments described here were excised from the same electrophoresis lane.

The use of plasmids is also a crucial factor in the success of this computation. Knapsack items could also be encoded in simple DNA elements, without a vector. However, the plasmid provides better control of the library generation and allows for reliable amplification

and storage. The plasmid method itself can also be used in scaling to larger instances, as inserts over 10^4 bp are quite common. If elements on the same scale as used here are inserted, a single plasmid could easily accommodate hundreds of knapsack elements. In practice, however, before such large insert sizes can be attained, the number of commercially available restriction endonucleases, as well as the number of theoretically available enzyme recognition sites will become prohibitive.

A more important factor in scaling is probably the separation technology used. Polyacrylamide gel electrophoresis is, in theory, capable of separating DNA fragments with a resolution of 0.1%, i.e. discrimination between 999 and 1000 bp fragments is feasible (Sambrook and Russell, 2001). However, these figures are for dedicated full length sequencing gels and capillary sequencing systems. Here, 25 cm gels were used in the selection phase, with the entire fragment library separated over approximately 10 cm. The smallest bands that can be excised are of the order of 1 mm long. Consequently, exact numerical solutions are not to be expected here, and indeed fragments differing up to 21 bp were recovered from the same slice. Still, as an enrichment procedure, preparative gel electrophoresis is a promising system: for knapsack instances with capacities corresponding to up to several tens of kilobases, the library to be searched can be narrowed down considerably. Exact solutions can then be obtained by sequencing of clones or microarray analysis.

These limitations imply that the plasmid methodology outlined here will be insufficient for solving knapsack problems of interesting dimensions. However, this is a fundamental characteristic of all brute force algorithms for such problems, molecular or not. A possible way out of this predicament is the implementation of molecular evolutionary algorithms (Chen and Wood, 2000; Bäck et al., 2003). Such computations would start with an incomplete molecular library, and attempt to improve this through iterated selection and diversification (using *in vitro* evolution technology). Evolutionary DNA computing may be feasible for knapsack problems, as these allow for a straightforward fitness measure and selection procedure (Henkel and Kok, 2005).

The extension of the computation to include protein optimization is at the moment primarily of theoretical interest. The precise analysis of proteins is currently more technically challenging than DNA characterization. Also, formal knapsack problems are hard to encode in proteins, as the definitions ask for natural number values. The only natural number that proteins can provide is the number of amino acids they consist of, which is information already present in the encoding DNA. In contrast, the amino acid molecular weights used here are distributions over real numbers. Even so, the real interest of knapsack problems is not in formal definitions, but in physical occurrences, where natural numbers may or may not be applicable. One of these real world knapsack problems lies very close to the abstract computations implemented here: protein design. In this case, a complete protein is identified with a knapsack, and amino acids or protein domains with the items. The optimization function (in itself still poorly understood) of course considers far more complex properties than just molecular weight, and the knapsack capacity is not as explicit as in the formal case. Still, boundary values can be identified: proteins above a certain size may not pass cellular membrane pores, and infinitely long genes tend to be expressed at infinitely low rates. In this view, the molecular implementation of knapsack problems reported here can be considered a simplified case of directed protein evolution.

4. Materials and methods

4.1. Plasmid construction

pKnapsack1 was constructed by ligation of a synthetic oligonucleotide linker (Isogen Bioscience, Maarssen, NL) into the *Sal*I/*Avr*II of plasmid pMP6110 (Henkel et al., 2005). The linker replaced the pMP6110 *enod40* encoding region with a VSV-G encoding region (Roche Diagnostics) flanked by *Sal*I recognition sites and a fragment flanked by *Bgl*II recognition sites. Thus, the number of computational stations is increased to eight (see Table 1). The stop codon is 12 bp downstream of the last *Bgl*II site. The entire open reading frame (ORF) containing the computational stations is under the control of a pET9d derived T7 promoter and a consensus ribosome binding site (Fig. 1). Integrity of ORF and promoter were checked by DNA sequencing (Baseclear, Leiden, NL). Other features of pKnapsack1, a pOK12 derivative (Vieira and Messing, 1991), are a p15A *E. coli* origin of replication and a kanamycin resistance marker.

4.2. Plasmid modification

Stations were removed by digestion of 1–10 μg (determined by absorbance at 260 nm) plasmid with the appropriate enzyme (New England Biolabs) under recommended conditions. Linearized vectors were isolated from a methylene blue stained 1% TAE agarose gel and purified using QIAquick gel extraction columns (Qiagen). Ligations were then performed overnight at 16 °C using T4 DNA ligase (Roche), and the ligated plasmid was transformed to chemically competent *E. coli* XL1-blue (Sambrook and Russell, 2001). This procedure ensures the removal of the station from all plasmids, and selects for fully functional plasmids. Plasmids were isolated from bacteria using QIAprep plasmid purification kits (Qiagen).

4.3. Protein purification

For protein expression, the plasmid mixture was transformed to chemically competent *E. coli* BL21(DE3). This strain carries the T7 RNA polymerase gene under the control of the *lacUV5* promoter. Expressed computational protein was purified by Ni^{2+}/histidine affinity chromatography. Cells were grown to an OD_{600} of 0.5 in LB medium, after which T7 protein expression was induced by addition of isopropyl-β-D-thiogalactopyranoside (IPTG) to a final concentration of 1 mM. After 2 h, cells were harvested by centrifugation and lysed in urea buffer (8 M urea, 100 mM NaH_2PO_4, 10 mM Tris–Cl, pH 8.0). Fifty microliters of Ni-NTA agarose (Qiagen) was added to 500 μl of cleared lysate (corresponding to 5 ml of culture), and the mixture was incubated with shaking at 4 °C for 30 min. The resin was washed twice with 500 μl urea buffer at pH 6.3, and protein was eluted thrice with 50 μl urea buffer at pH 4.5.

4.4. Knapsack selection

*Hin*dIII/*Xba*I digested plasmid library was separated on a 25 cm 10% TBE polyacrylamide gel. The minor *Hin*dIII/*Xba*I fragment contains the entire computational region (Fig. 1). Bands of desired length were cut from the SYBR Gold (Molecular Probes, Leiden, NL) stained gel and isolated by overnight soaking at 4 °C in diffusion buffer (0.5 M NH_4Ac, 0.1% SDS, 2 mM EDTA), centrifugation in 0.22 μm Ultrafree-MC spin filters (Millipore) and ethanol precipitation. Fragments were then religated into the vector (major *Hin*dIII/*Xba*I fragment).

Acknowledgement

This work was financially supported by the Netherlands Organization for Scientific Research (NWO), Council for Exact Sciences.

References

Aoi, Y., Yoshinobu, T., Tanizawa, K., Kinoshita, K., Iwasaki, H., 1998. Solution of the knapsack problem by deoxyribonucleic acid computing. Jpn. J. Appl. Phys. Part 1 37, 5839–5841.

Bäck, T., Kok, J.N., Rozenberg, G., 2003. Evolutionary computation as a paradigm for DNA-based computing. In: Landweber, L.F., Winfree, E. (Eds.), Evolution as Computation. DIMACS Workshop. Princeton, January 1999. Springer-Verlag, Berlin, pp. 15–40.

Chang, W.-L., Ho, M.S.-H., Guo, M., 2004. Molecular solutions for the subset-sum problem on DNA-based supercomputing. Biosystems 73, 117–130.

Chen, J.H., Wood, D.H., 2000. Computation with biomolecules. Proc. Natl. Acad. Sci. U.S.A. 97, 1328–1330.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability. A Guide to the Theory of NP-completeness. W.H. Freeman and Company, New York.

Head, T., 2000. Circular suggestions for DNA computing. In: Carbone, A., Gromov, M., Prusinkiewcz, P. (Eds.), Pattern Formation in Biology, Vision and Dynamics. World Scientific, Singapore, pp. 325–335.

Head, T., Rozenberg, G., Bladergroen, R.S., Breek, C.K.D., Lommerse, P.H.M., Spaink, H.P., 2000. Computing with DNA by operating on plasmids. Biosystems 57, 87–93.

Head, T., Chen, X., Nichols, M.J., Gal, S., 2002. Aqueous solutions of algorithmic problems: emphasizing knights on a 3 × 3. In: Jonoska, N., Seeman, N.C. (Eds.), DNA Computing, Proceedings of the Seventh International Meeting on DNA Based Computers. Springer-Verlag, Berlin, pp. 191–202.

Henkel, C.V., Bladergroen, R.S., Balog, C.I.A., Deelder, A.M., Head, T., Rozenberg, G., Spaink, H.P., 2005. Protein output for DNA computing. Nat. Comput. 4, 1–10.

Henkel, C.V., Kok, J.N., 2005. Towards evolutionary DNA computing. In: Mira, J., Álvarez, J.R. (Eds.), Mechanisms, Symbols, and Models Underlying Cognition: First International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2005. Springer-Verlag, Berlin, pp. 242–257.

Paun, G., Rozenberg, G., Salomaa, A., 2005. DNA Computing—New Computing Paradigms. Springer-Verlag, Berlin.

Pérez-Jiménez, M.J., Sancho-Caparrini, F., 2002. Solving knapsack problems in a sticker based model. In: Jonoska, N., Seeman, N.C. (Eds.), DNA Computing, Proceedings of the Seventh International Meeting on DNA Based Computers. Springer-Verlag, Berlin, pp. 161–171.

Sambrook, J., Russell, D.W., 2001. Molecular Cloning: A Laboratory Manual. Cold Spring Harbor, New York.

Stoschek, E., Sturm, M., Hinze, T., 2001. DNA-computing—ein funktionales Modell im laborpraktischen Experiment. Informatik Forsch. Entw. 16, 35–52.

Vieira, J., Messing, J., 1991. New pUC-derived cloning vectors with different selectable markers and DNA-replication origins. Gene 100, 189–194.