UNIVERSITY OF OXFORD

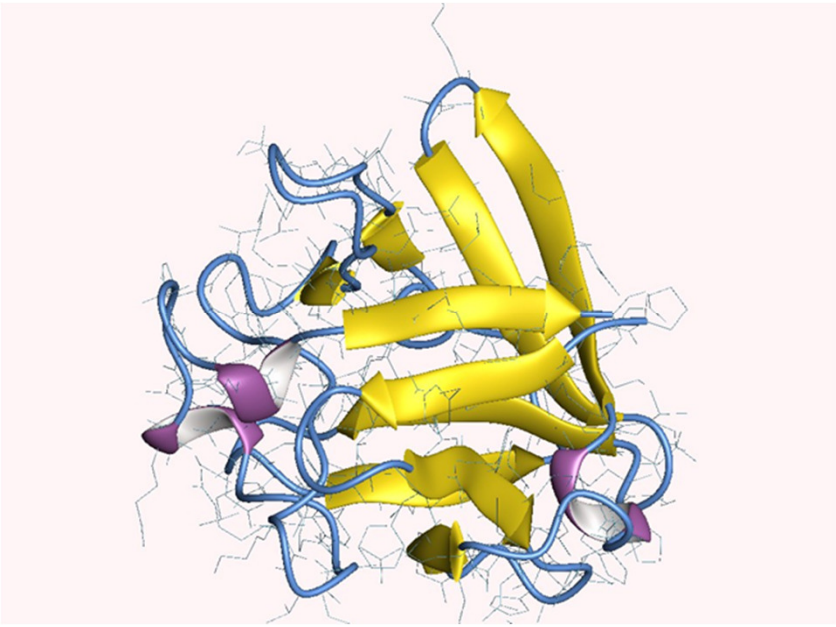
Computing Reliably with Molecular Walkers

Marta Kwiatkowska, University of Oxford

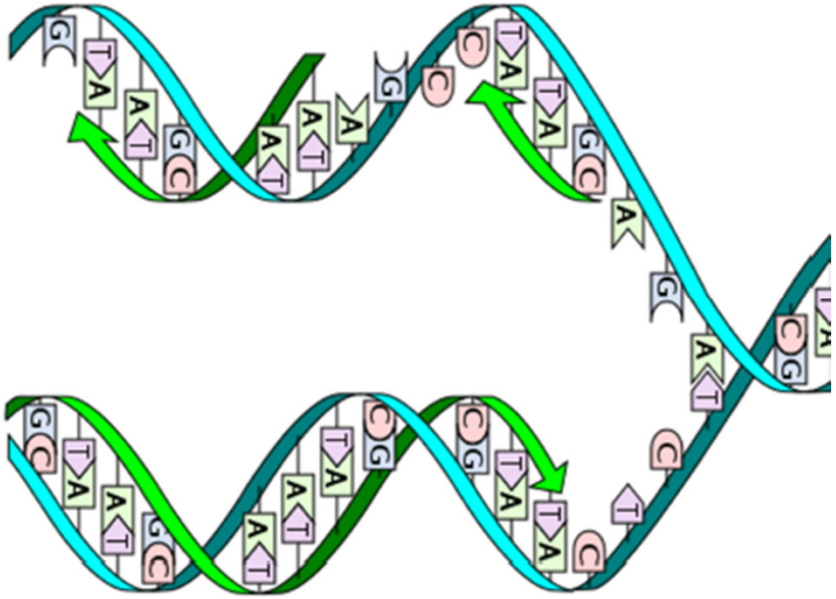
NWPT 2015, Reykjavik

At the nanoscale…

- The world of molecules



Human FGF protein



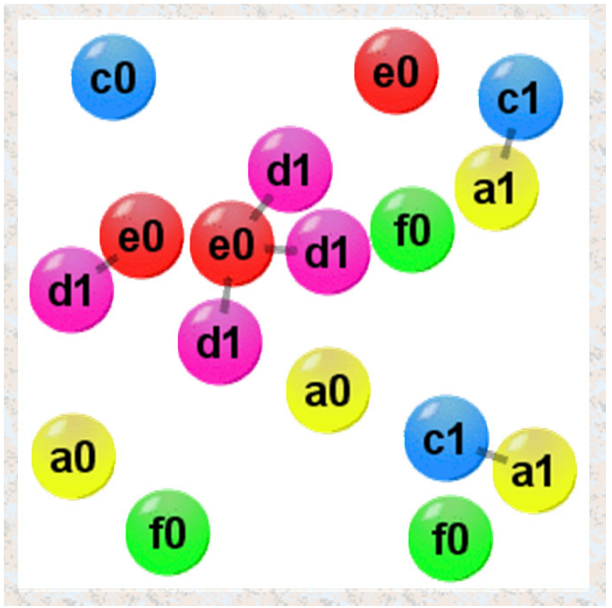
width 2nm

DNA: versatile, easy to synthesize

Molecular programming

- The application of computational concepts and design methods to nanotechnology, esp biochemical systems

- Molecular programs are
  - networks of molecules
  - can interact
  - can move
  - can self-assemble



- Key observation
  - can store/process information
  - are programmable
  - (can compute a desired outcome)
  - proceed autonomously

- Need programming languages, modelling, verification, …

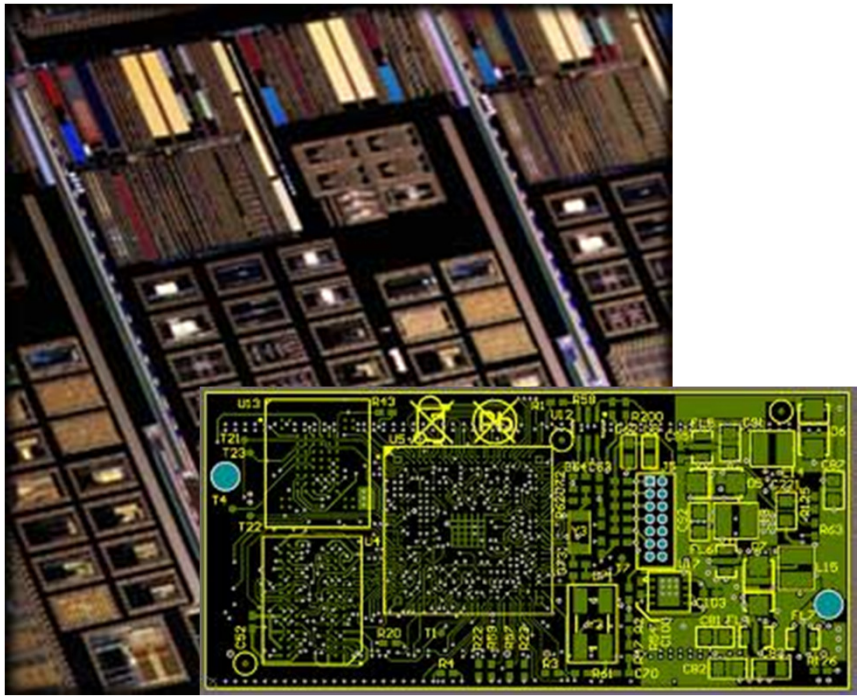
What is a molecular program?

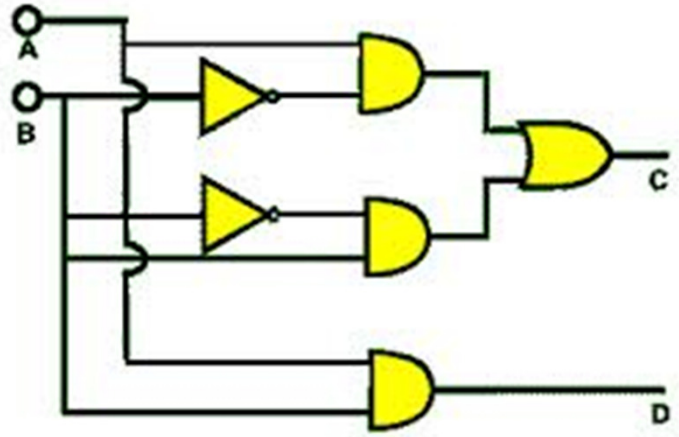
- A set of chemical reactions…

$$A + B \xrightarrow{k_1} C + D \qquad A + C \xrightarrow{k_2} E$$

- A chemical reaction network (CRN)
- Computing with chemistry!
- Important fact: any finite CRN can be implemented with DNA molecules!
- DNA used as information processing material
- Several technologies exist: DNA Strand Displacement (DSD)

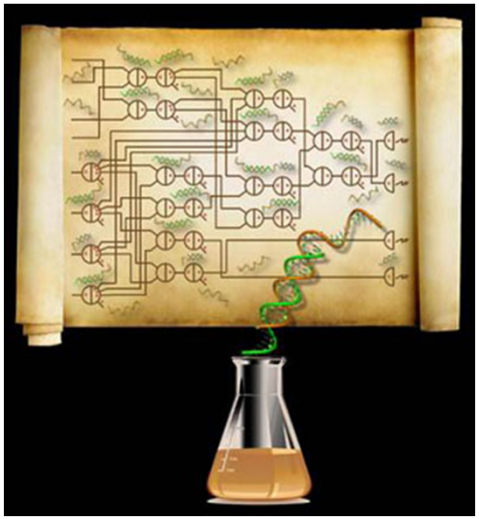
Digital circuits





- Logic gates realised in silicon
- 0s and 1s are represented as low and high voltage
- Hardware verification indispensable as design methodology

DNA circuits, in solution

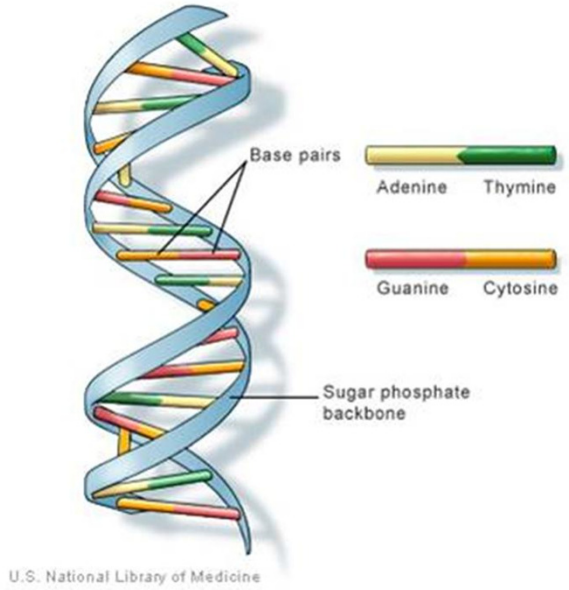


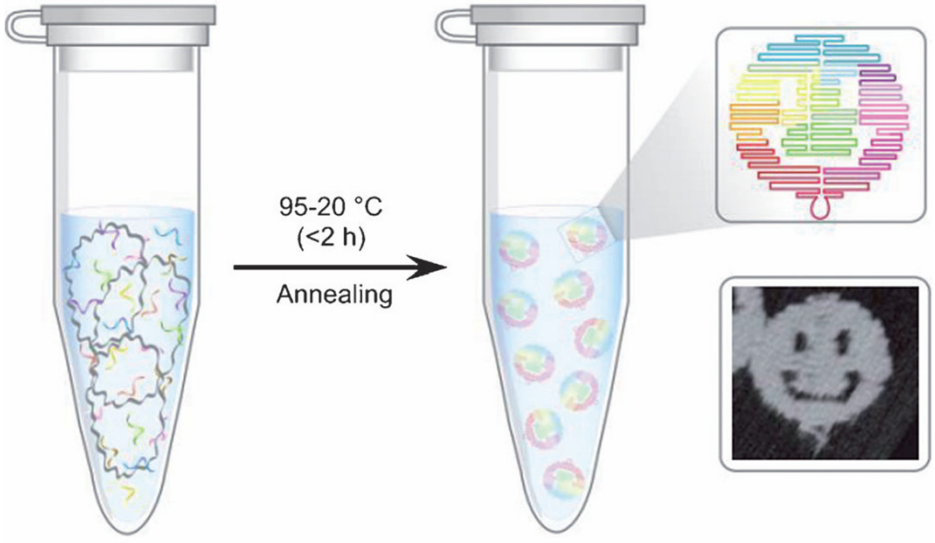
[Qian, Winfree, *Science* 2012]

- "Computing with soup" (The Economist 2012)
- Single strands are inputs and outputs
- Circuit of 130 strands computes square root of 4 bit number, rounded down
- 10 hours, but it's a first…

Pop quiz, hotshot: what's the square root of 13?
Science Photo Library/Alamy

DNA nanostructures



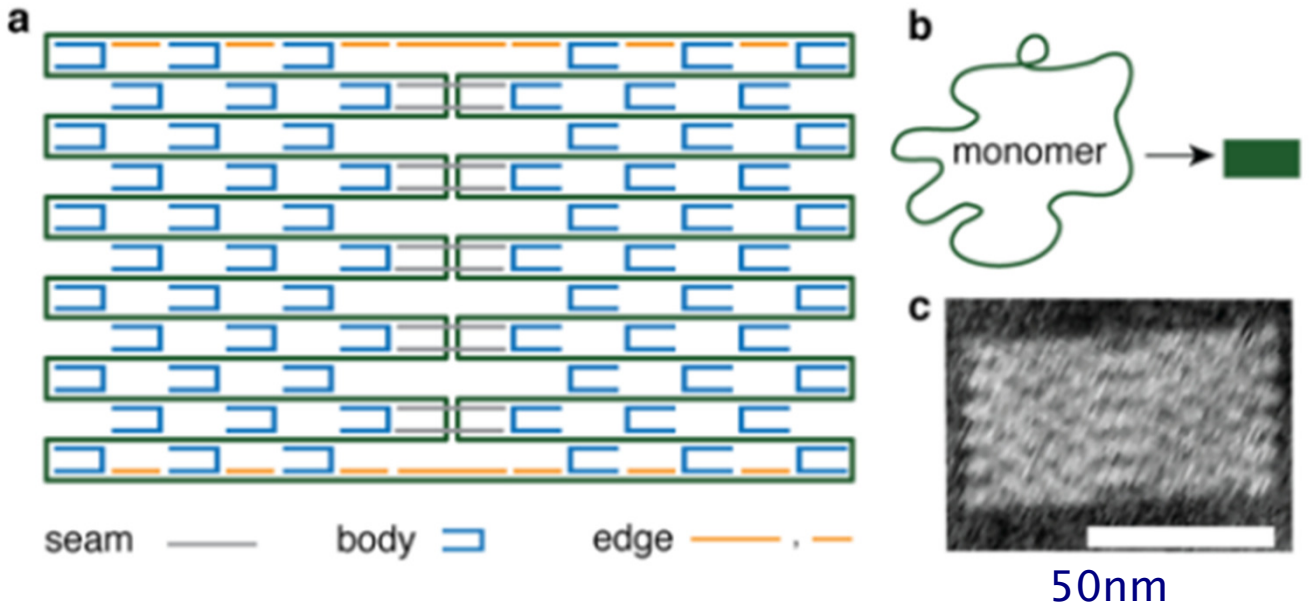


DNA origami

- DNA origami [Rothemund, *Nature* 2006]
  - DNA can self-assemble into structures – "molecular IKEA?"
  - programmable self-assembly (can form tiles, nanotubes, boxes that can open, etc)
  - simple manufacturing process (heating and cooling), not yet well understood

DNA origami tiles

- Origami tiles made from DNA [Turberfield lab]

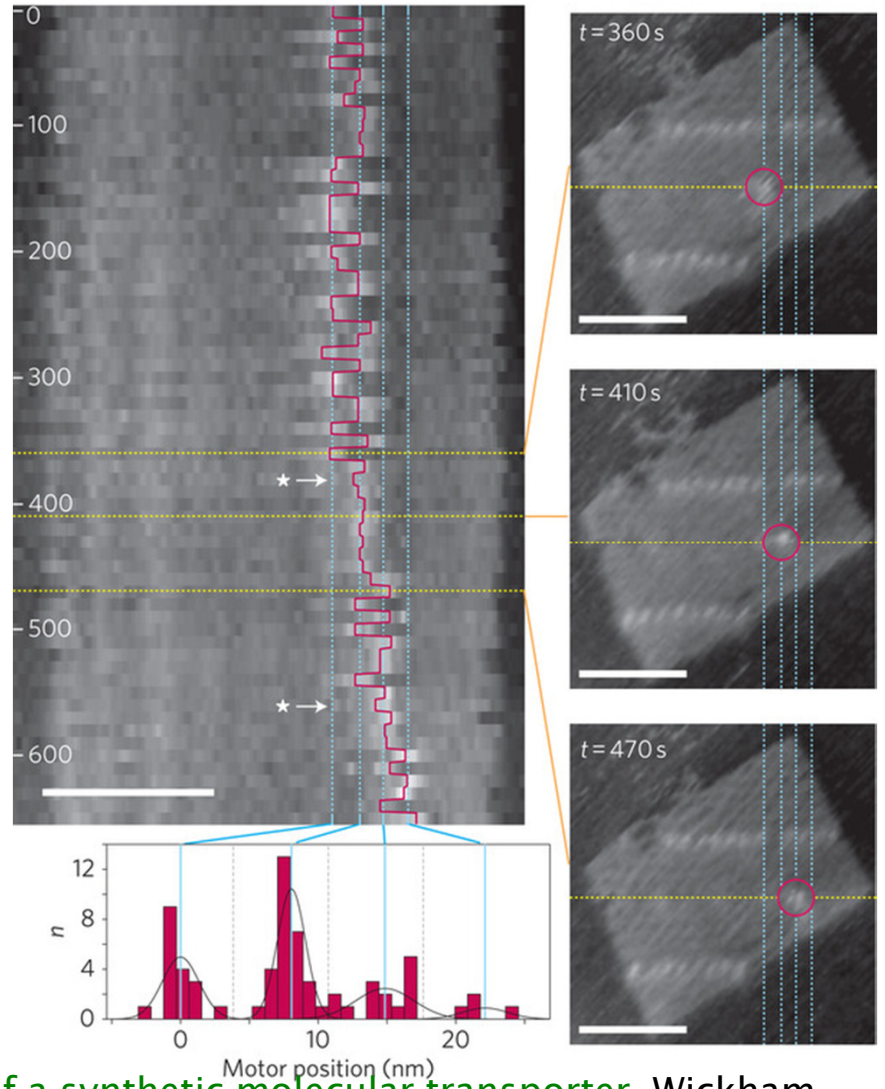


a. Tile design, showing staples 'pinning down' the monomer and highlighting seam staples
b. Circular single strand that folds into tile
c. AFM image of the tile

Guiding the folding pathway of DNA origami. Dunne, Dannenberg, Ouldridge, Kwiatkowska, Turberfield & Bath, Nature (in press)

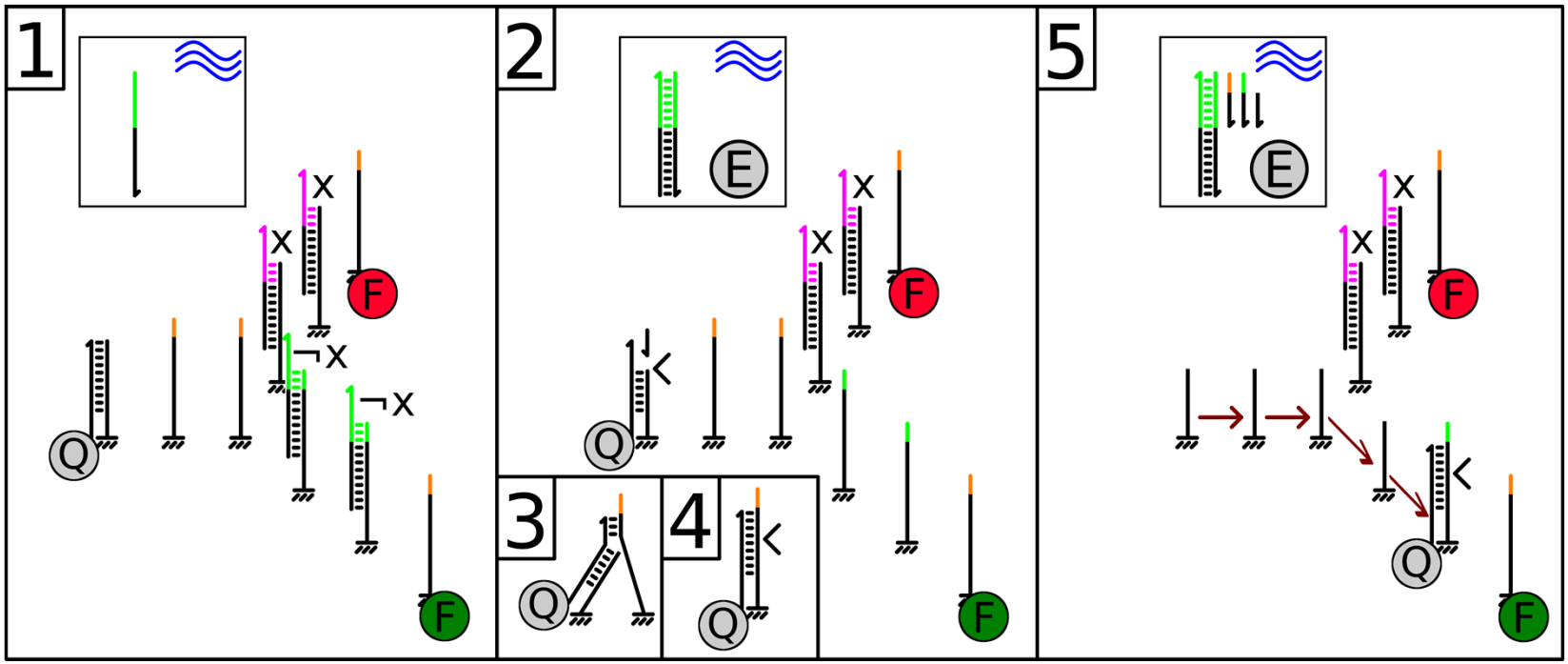
DNA walkers

- How it works…
  - tracks made up of anchor strands laid out on DNA origami tile
  - can make molecule 'walk' by attaching/ detaching from anchor
  - autonomous, constant average speed
  - can control movement
  - can carry cargo
  - all made from DNA



Direct observation of stepwise movement of a synthetic molecular transporter. Wickham *et al*, Nature Nanotechnology 6, 166–169 (2011)

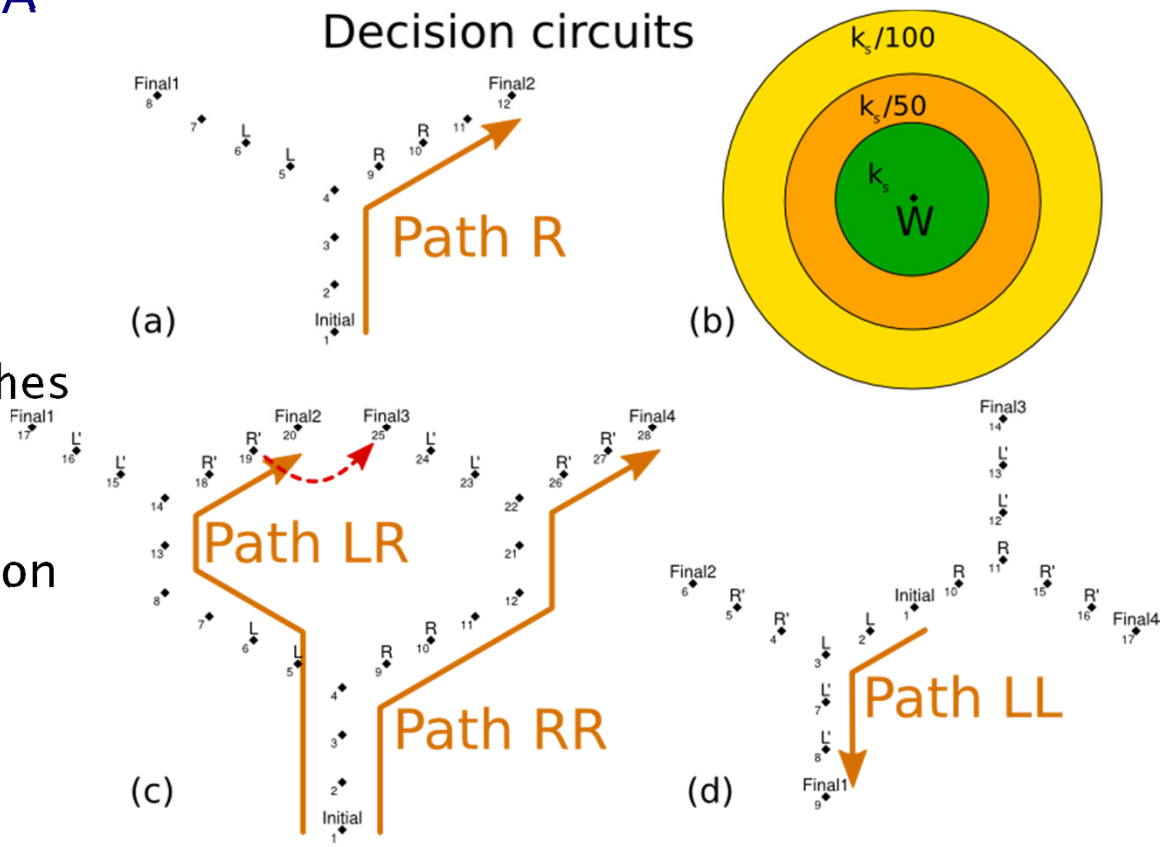
Walker stepping action in detail…



1. Walker carries a quencher (Q)
2. Sections of the track can be selectively unblocked
3. Walker detaches from anchor strand
4. Walker attaches to the next anchor along the track
5. Fluorophores (F) detect walker reaching the end of the track

DNA walker circuits

- Computing with DNA walkers
  - branching tracks laid out on DNA origami tile
  - starts at ‘initial’, signals when reaches ‘final’
  - can control ‘left’/’right’ decision
  - (this technology) single use only, ‘burns’ anchors



- Localised computation, well mixed assumption as in solution does not apply

Why DNA programming?

- DNA: versatile, easily accessible, cheap to synthesise material
- Biocompatible, good for biosensors
  - programmable identification of substance, targeted delivery
- Moore's law, hence need to make devices smaller…
  - DNA computation, directly at the molecular level
  - nanorobotics, via programmable molecular motion
- Many applications for combinations of DNA logic circuits, origami and nanorobotics technologies
  - e.g. point of care diagnostics, smart therapeutics, …

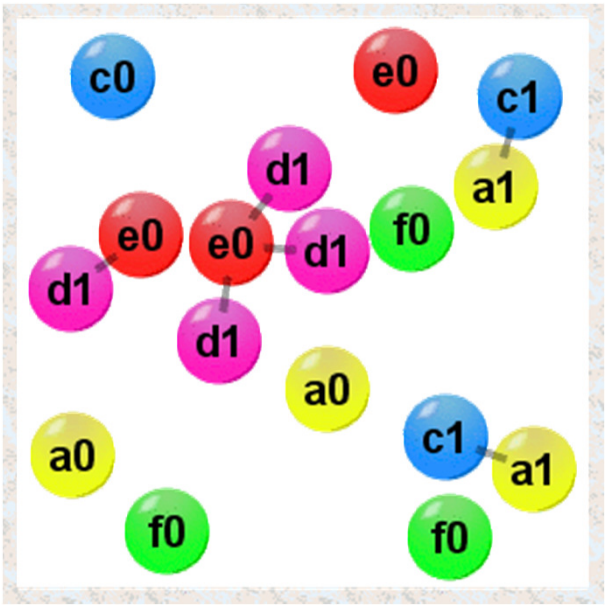
- What good is quantitative verification in this application domain?
  - stochasticity essential!
  - reliability of computation is an issue

This lecture…

- Quantitative modelling and verification for molecular programming
  - probabilistic model checking and PRISM
- Lessons learnt
  - automatic debugging DNA computing devices
  - analysing reliability of molecular walkers
  - not just verification: can we automatically synthesise reaction rates to guarantee a specified level of reliability?
  - can we analyse the origami folding process and make predictions?
- Challenges and directions

Modelling molecular networks

- Focus on modelling dynamics and analysis of behaviours
  - networks of molecules
  - molecular interaction
  - molecular motion
  - self-assembly



- Rather than
  - geometry
  - structure
  - sequence

- Chemical reaction networks

- Emphasis on quantitative/probabilistic characteristics
- Stochasticity essential for low molecular counts

Chemical reaction networks

Used to encode a molecular mechanism

1: FGF binds/releases FGFR

FGFR + FGF $\rightarrow$ FGFR:FGF $\quad k_1 = 5e+8\ M^{-1}s^{-1}$

FGFR + FGF $\leftarrow$ FGFR:FGF $\quad k_2 = 0.002\ s^{-1}$

2: Relocation of FGFR (whilst phosphorylated)

FGFR $\rightarrow$ $\quad k_3 = 0.1\ s^{-1}$

Can map to different semantics/representation

Chemical reaction networks

Used to encode a real or hypothetical mechanism

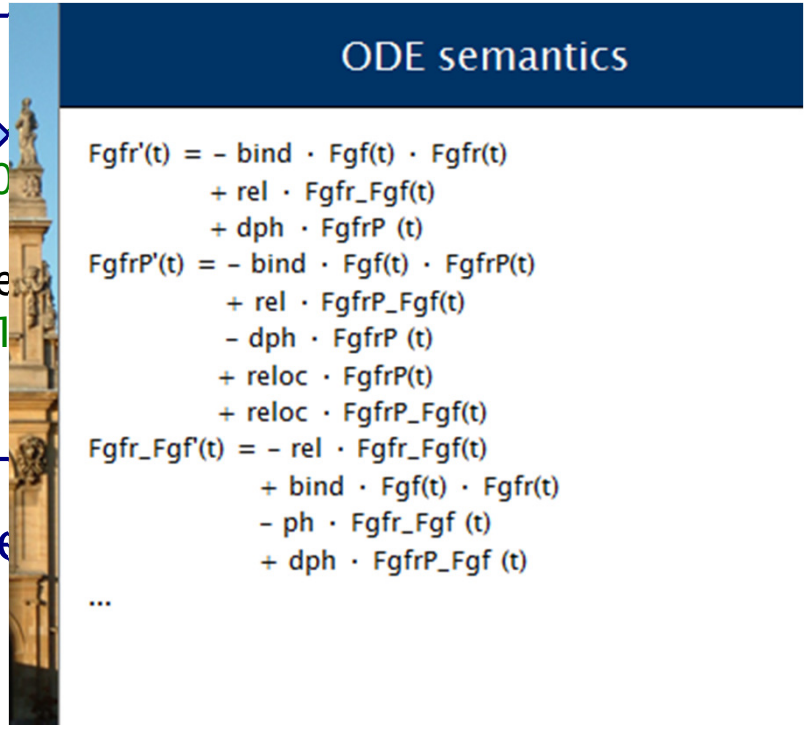
1: FGF binds/releases FGFR

FGFR + FGF → FGFR:FG

FGFR + FGF ← FGFR:FGF k_2=0.0

2: Relocation of FGFR (whilst phosphorylate

FGFR → k_3=0.1



ODE semantics

$$
\begin{aligned}
Fgfr'(t) &= -bind \cdot Fgf(t) \cdot Fgfr(t) \\
&\quad + rel \cdot Fgfr\_Fgf(t) \\
&\quad + dph \cdot FgfrP(t) \\
FgfrP'(t) &= -bind \cdot Fgf(t) \cdot FgfrP(t) \\
&\quad + rel \cdot FgfrP\_Fgf(t) \\
&\quad - dph \cdot FgfrP(t) \\
&\quad + reloc \cdot FgfrP(t) \\
&\quad + reloc \cdot FgfrP\_Fgf(t) \\
Fgfr\_Fgf'(t) &= -rel \cdot Fgfr\_Fgf(t) \\
&\quad + bind \cdot Fgf(t) \cdot Fgfr(t) \\
&\quad - ph \cdot Fgfr\_Fgf(t) \\
&\quad + dph \cdot FgfrP\_Fgf(t) \\
&\ldots
\end{aligned}
$$

Chemical reaction networks

Used to encode a real or hypothetical mechanism

1: FGF binds/releases FGFR

FGFR + FGF → FGFR:FG

FGFR + FGF ← FGFR:FGF k_2=0.0

2: Relocation of FGFR (whilst phosph

FGFR →

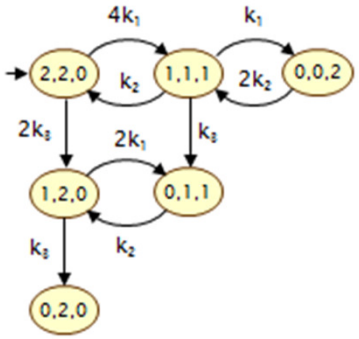
Can map to different semantics/

ODE semantics

Fgfr'(t) = – bind · Fgf(t) · Fgfr(t)
+ rel · Fgfr_Fgf(t)

CTMC semantics

- CTMC with state space
  - (|FGFR|,|FGF|,|FGFR:FGF|)
  - initially (2,2,0)



- under assumption of mass action kinetics

8

Chemical reaction networks

Used to encode a real or hypothetical mechanism

1: FGF binds/releases FGFR

FGFR + FGF → FGFR:FG

FGFR + FGF ← FGFR:FGF $k_2=0.0$

2: Relocation of FGFR (whilst phosph

FGFR →

Can map to different

ODE semantics

Fgfr'(t) = – bind · Fgf(t) · Fgfr(t)
+ rel · Fgfr_Fgf(t)

CTMC semantics

- CTMC with state space
  - (|FGFR|,|FGF|,|FGFR:FGF|)
  - initially (2,2,0)

PRISM reactive modules

```
module fgfr

  fgfr  : [0..1] init 0; // 0 - free, 1 - bound
  phos  : [0..1] init 0; // 0 - unphosphorylated, 1 - phosphorylated
  reloc : [0..1] init 0; // 0 - not relocated, 1- relocated

  [bnd] reloc=0 ∧ fgfr=0 → k1 : (fgfr'=1); // FGF and FGFR bind
  [rel]  reloc=0 ∧ fgfr=1 → k2 : (fgfr'=0); // FGF and FGFR release
  [] reloc=0 ∧ fgfr =1 ∧ phos =0 → k3 : (phos'=1); // FGFR phosphor.
  [] reloc=0 ∧ phos=1     → k4 : (phos'=0); // FGFR dephosphorylates
  [] reloc=0 ∧ phos=1     → k5 : (reloc'=1); // FGFR relocates

endmodule
```

8

Chemical reaction networks

Used to encode a real or hypothetical mechanism

1: FGF binds/releases FGFR

FGFR + FGF → FGFR:FG

FGFR + FGF ← FGFR:FGF $k_2=0.0$

2: Relocation of FGFR (whilst phosph

FGFR →

Can map to different

ODE semantics

Fgfr'(t) = – bind · Fgf(t) · Fgfr(t)
+ rel · Fgfr_Fgf(t)

CTMC semantics

- CTMC with state space
  - (|FGFR|,|FGF|,|FGFR:FGF|)
  - initially (2,2,0)

PRISM reactive modules

module fgfr

SBML code

```
<listOfSpecies>
  <species id="FGFR" initialConcentration="1" ... />
  <species id="FGF" initialConcentration="1" ... /> ...
</listOfSpecies>
<reaction id="Reaction1" reversible="true">
  <listOfReactants>
    <speciesReference species="FGFR" />...
  </listOfReactants>
  <listOfProducts>
```

Chemical reaction networks

Used to encode a real or hypothetical mechanism

1: FGF binds/releases FGFR

FGFR + FGF → FGFR:FGF k_1=5e+8 $M^{-1}s^{-1}$

FGFR + FGF ← FGFR:FGF k_2=0.002 s^{-1}

2: Relocation of FGFR (whilst phosphorylated)

FGFR → k_3=0.1 s^{-1}

Can map to different semantics/representation

- Now can apply probabilistic model checking to obtain model predictions…
  - software tools exist and are well used, e.g. PRISM
- Sounds easy?

The PRISM model checker

- Inputs CTMC models in reactive modules or SBML
- and specifications given in probabilistic temporal logic CSL
  - what is the probability that the concentration reaches min?

    $P_{=?}$ [F c≥min]
  - in the long run, what is the probability that the concentration remains stable between min and max?

    $S_{=?}$ [(c ≥min)∧(c≤max)]

- Then computes model predictions via
  - exhaustive analysis to compute probability and expectations over time (with numerical precision)
  - or probability estimation based on simulation (approximate, with confidence interval)
- See www.prismmodelchecker.org

PRISM 4.0:Verification of Probabilistic Real-time Systems, Kwiatkowska *et al*, In *Proc. CAV'11*

Quantitative probabilistic verification

- What's involved
  - specifying, extracting and building of quantitative models
  - model reduction
    - BDD/MTBDD, bisimulation quotient, adaptive aggregation
  - graph-based analysis: reachability + qualitative verification
    - symbolic (BDD) fixpoint computation
  - numerical solution, e.g. linear equations/linear programming
    - symbolic (MTBDD), explicit, sparse, hybrid
    - uniformisation, fast adaptive uniformisation
  - simulation-based statistical model checking
    - Monte Carlo, estimation (confidence interval), hypothesis testing
- Typically computationally more expensive

Historical perspective

- First use of PRISM for modelling molecular networks in 2005
  - [Calder, Vyshemirsky, Gilbert and Orton, …]
  - RKIP inhibited ERK pathway
- 2006 onwards: PRISM enhanced with SBML import
  - predictive modelling of the FGF pathway [Heath, Kwiatkowska, Norman, Parker and Tymchyshyn]
  - predictions experimentally validated [Sandilands et al, 2007]
- Since 2012 PRISM has been applied to DNA computation
  - PRISM connected to Microsoft's Visual DSD (DNA computing design tool) [Lakin, Parker, Cardelli, Kwiatkowska and Phillips]
  - expressiveness and reliability of DNA walker circuits studied [Dannenberg, Kwiatkowska, Thachuk, Turberfield]
- Scalability of PRISM analysis limited

Three DNA case studies

Applying quantitative modelling, verification and synthesis to three DNA case studies

1. DNA tranducer gate design (with Cardelli)
2. DNA walker design (with Turberfield lab)
3. DNA origami dimer (with Turberfield lab)

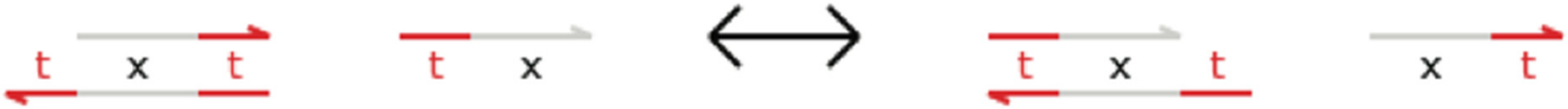
All CTMC models, 1&2 modelled in PRISM

Lessons learnt…

1. Cardelli's DNA transducer gate

- DNA computing with a restricted class of DNA strand displacement structures (process algebra by Cardelli)
  - double strands with nicks (interruptions) in the top strand

  - and two-domain single strands consisting of one toehold domain and one recognition domain

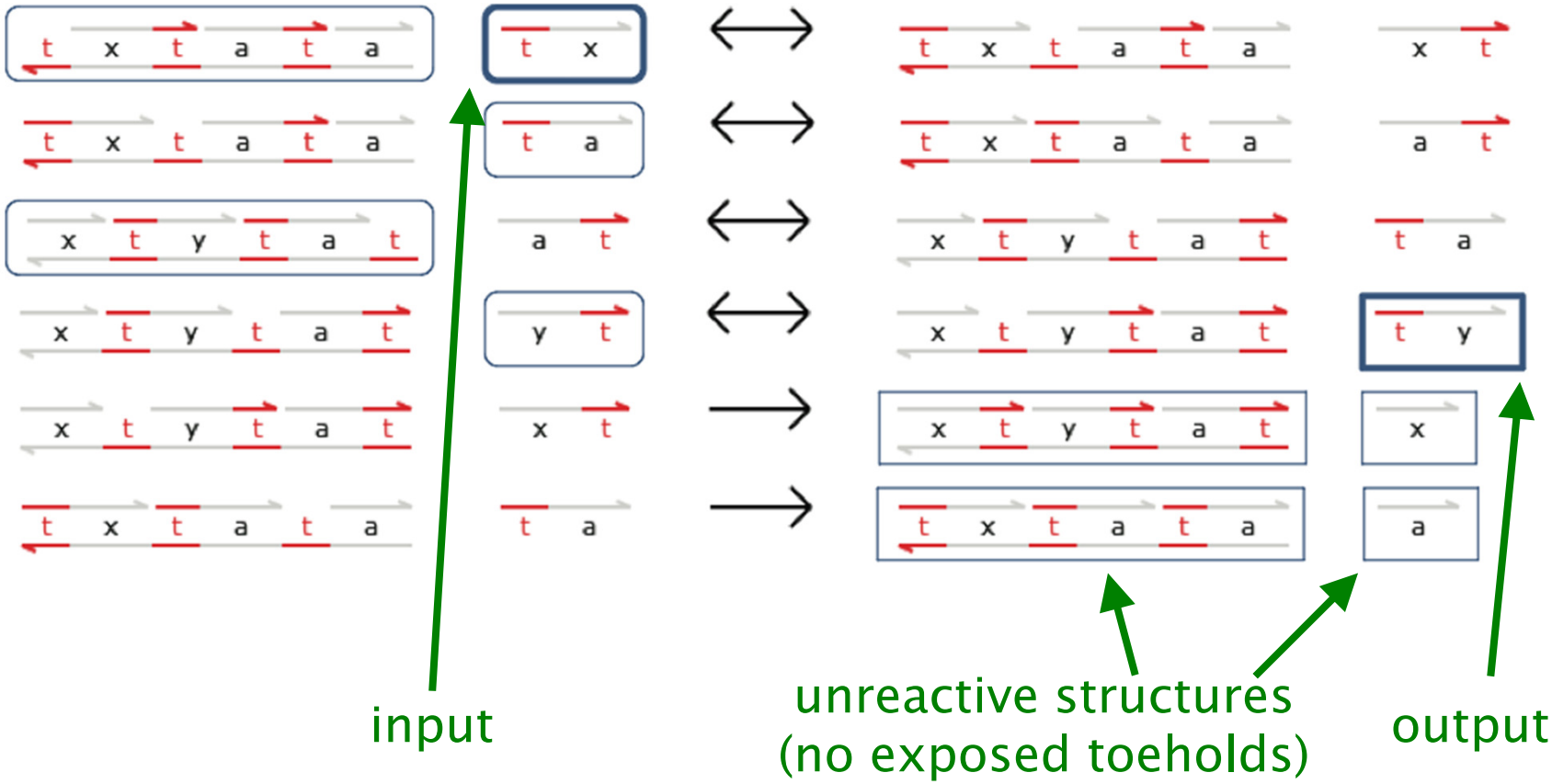


  - "toehold exchange": branch migration of strand <t^ x> leading to displacement of strand <x t^>
- Used to construct transducers, fork/join gates
  - which can emulate Petri net transitions
  - can be formed into cascades [Qian, Winfree, *Science* 2011]

Two-Domain DNA Strand Displacement. Cardelli, L. *Proc. Development of Computational Models (DCM'10)*, 2010

Transducer example

- Transducer: full reaction list

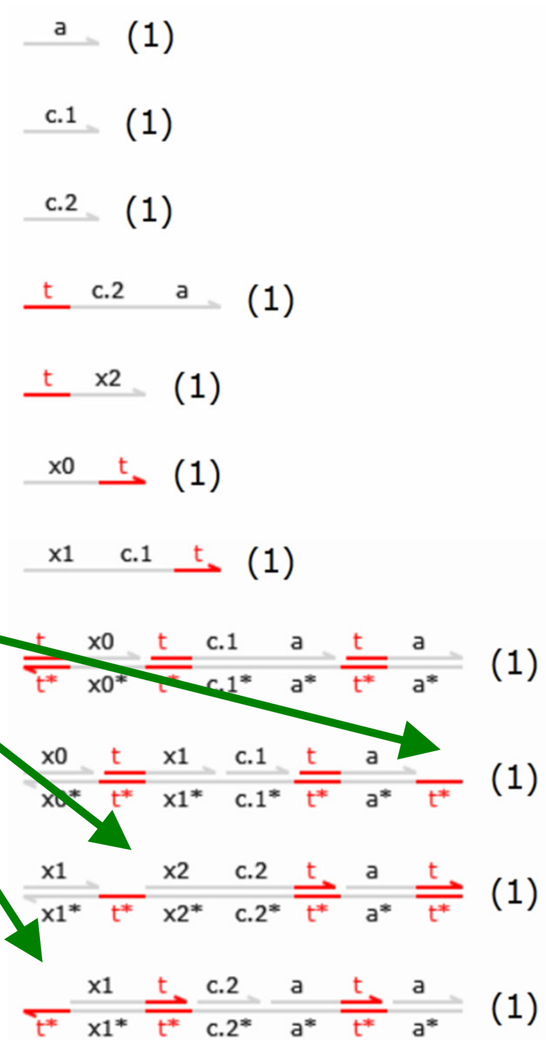


Transducers: correctness

- Formalising correctness…
  - identify states where gate has terminated correctly: "all_done”
  - (correct number of outputs, no reactive gates left)
- Check:
  - (i) any possible deadlock state that can be reached must satisfy "all_done”
    (ii) there is at least one path through the system that reaches a state satisfying "all_done”
- In temporal logic (CTL):
  - A [G "deadlock" => "all_done"]
  - E [F "all_done"]
- Verifies using PRISM (back end to Visual DSD)…
  - for one transducer: both properties true
  - for two transducers in series: (ii) is true, but (i) is false

DNA transducer flaw

- Cardelli's DNA transducer gate
  - inputs/outputs single strands
  - can be connected into cascades
- PRISM identifies a bug: 5-step trace to a "bad" deadlock state
  - previously found manually [Cardelli'10]
  - detection now fully automated
- Bug is easily fixed
  - (and verified)



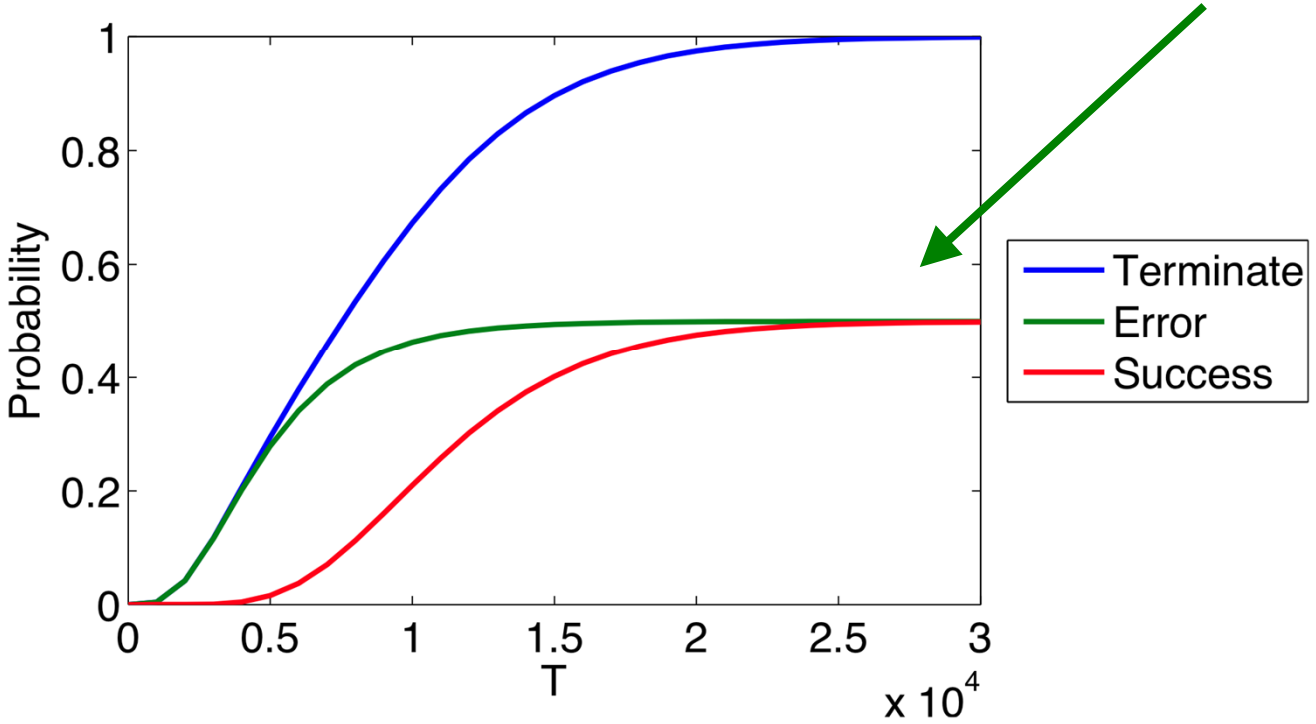
Counterexample:
(1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,1,1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,1,1,0,0,1,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,0,1,0,0,1,1,1,0,0,0,1,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,0,1,0,0,1,0,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0)

Design and Analysis of DNA Strand Displacement Devices using Probabilistic Model Checking, Lakin *et al*, Journal of the Royal Society Interface, 9(72), 1470-1485, 2012

Quantitative properties

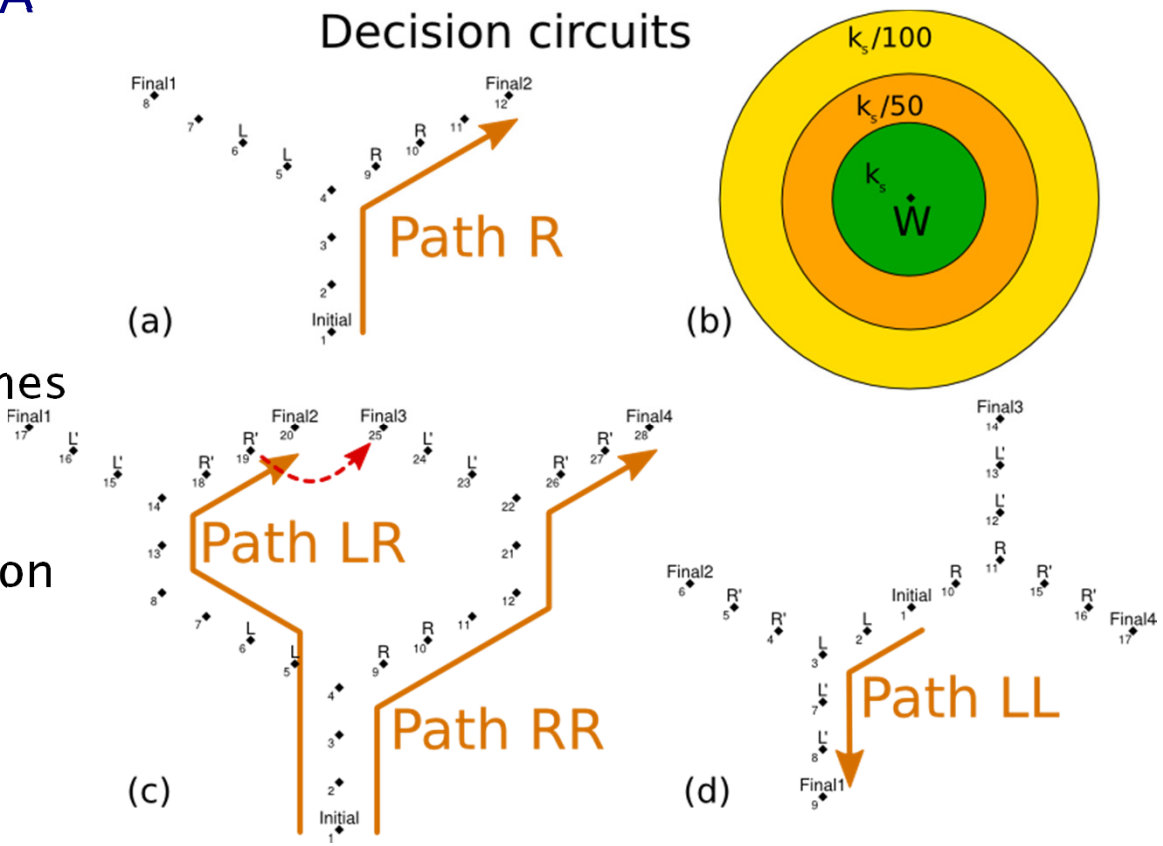
- We can also use PRISM to study the kinetics of the pair of (faulty) transducers:
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock"]
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock" & !"all_done"]
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock" & "all_done"]

success/error equally likely



2. DNA walker circuits

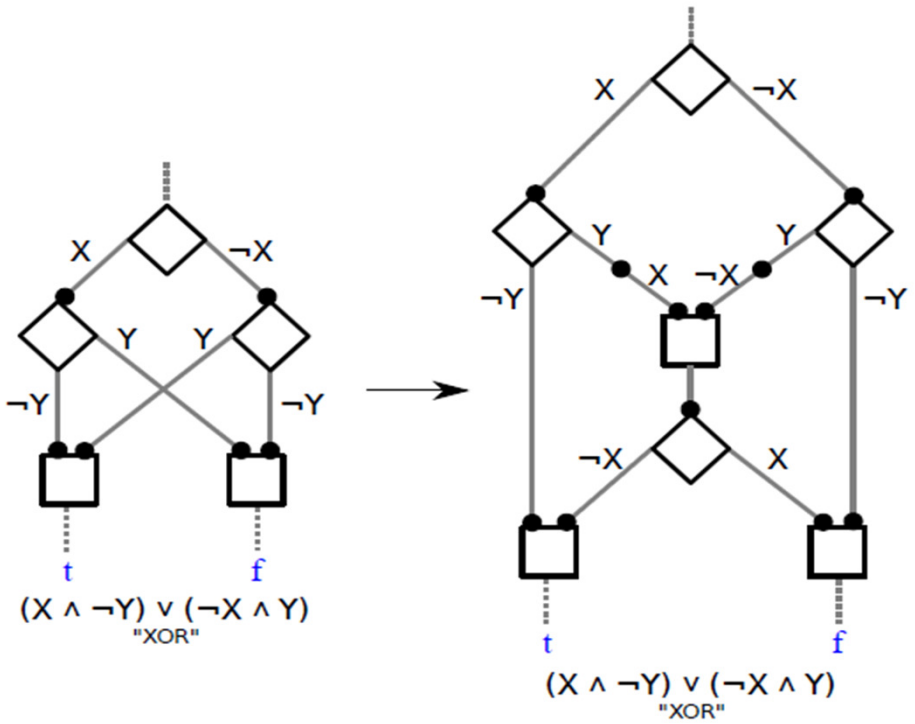
- Computing with DNA walkers
  - branching tracks laid out on DNA origami tile
  - starts at ‘initial’, signals when reaches ‘final’
  - can control ‘left’/’right’ decision
  - (this technology) single use only, ‘burns’ anchors



- But what can they compute?

DNA walkers: expressiveness

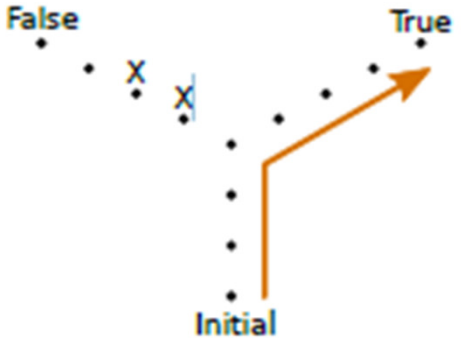
- Several molecular walker technologies exist
  - computation localised
  - faster computation times than in solution
- The 'burnt bridges' DNA walker technology
  - can compute any Boolean function
  - must be planar, needs rerouting
  - tracks undirected
  - reduction to 3-CNF, via a series of disjunction gates
  - limited parallel evaluation

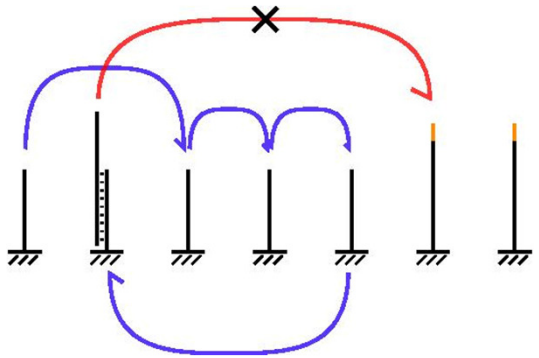


DNA walker circuits: Computational potential, design, and verification. Dannenberg *et al*, Natural Computing, To appear, 2014

DNA walkers: applications

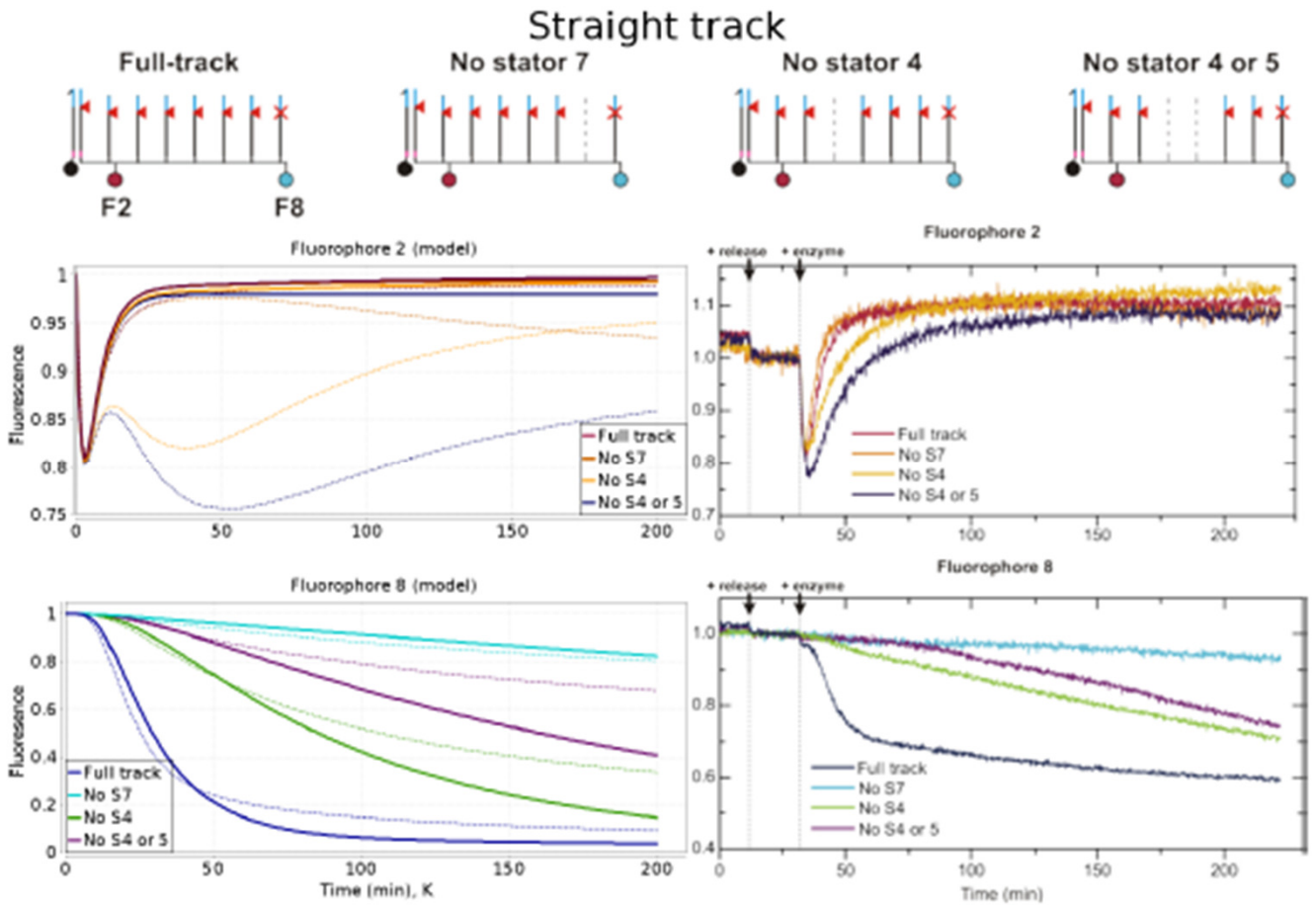
- Walkers can realise biosensors: safety/reliability paramount
- Molecular walker computation inherently unreliable…
  - 87% follow the correct path
  - can jump over one or two anchorages, can deadlock





- Analyse reliability of molecular walker circuits using PRISM
  - devise a CTMC model, fit to experimental data
  - analyse reliability, deadlock and performance
  - use model checking results to improve the layout

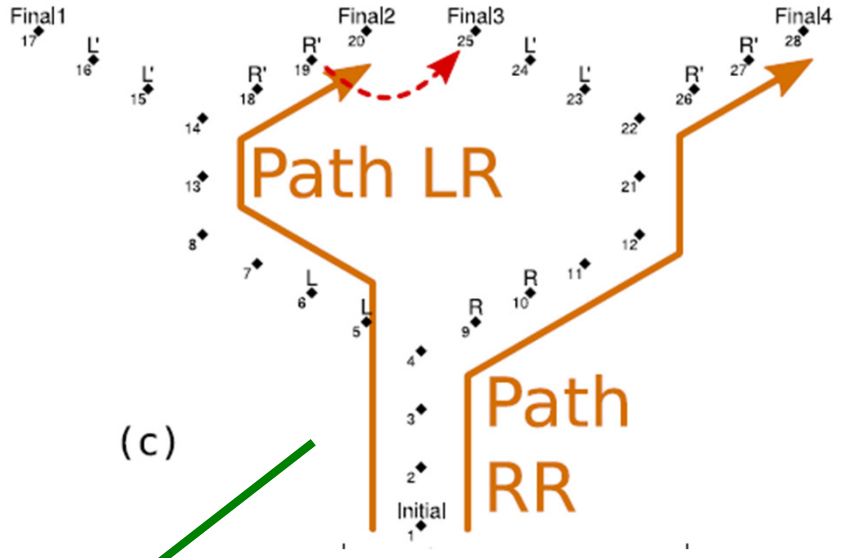
DNA walkers: model fitting



Fitting single-junction circuit to data (dotted lines alternative model)

DNA walkers: results

- Model predictions reasonably well aligned with experiments
- Results confirm effect of leak reactions
- Improve layout guided by model checking
- Can synthesise rates to guarantee reliability level

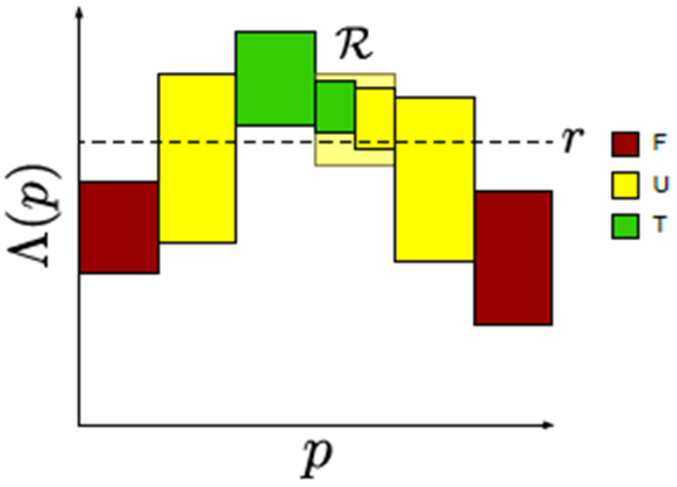


	Single-junction						Double-junction									
	Experiment			Model			Experiment				Model				Optimised (model)			
%	R	R^2	L/R	R	R^2	L/R	RR	RL	LR	LL	RR	RL	LR	LL	RR	RL	LR	LL
Finishes	65	56	56	97	96	92	33	40	22	33	90	89	89	90	94	92	94	92
Correct	76	87	50	78	85	50	70	65	55	76	77	74	74	77	78	78	78	78
Deadlock				.084	.16	.063					1.0	1.7	1.7	1.1	0.0	0.0	0.0	0.0
Steps				7.1	7.0	6.6					11.7	11.8	11.8	11.7	5.1	5.1	5.1	5.1

http://www.prismmodelchecker.org/casestudies/dna_walkers.php

From verification to synthesis…

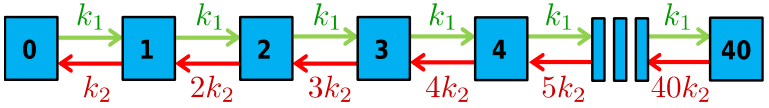
- Automated verification aims to establish if a property holds for a given model
- Can we find a model so that a property is satisfied?
  - difficult…
- The parameter synthesis problem is
  - given a parametric model, property and probability threshold
  - find a partition of the parameter space into True, False and Uncertain regions s.t. the relative volume of Uncertain is less or equal than a given ϵ
- Successive region refinement, based on over & under approx., implemented in PRISM



Precise Parameter Synthesis for Stochastic Biochemical Systems. Ceska *et al*, In Proc. CMSB, LNCS, 2014

Example: satisfaction function

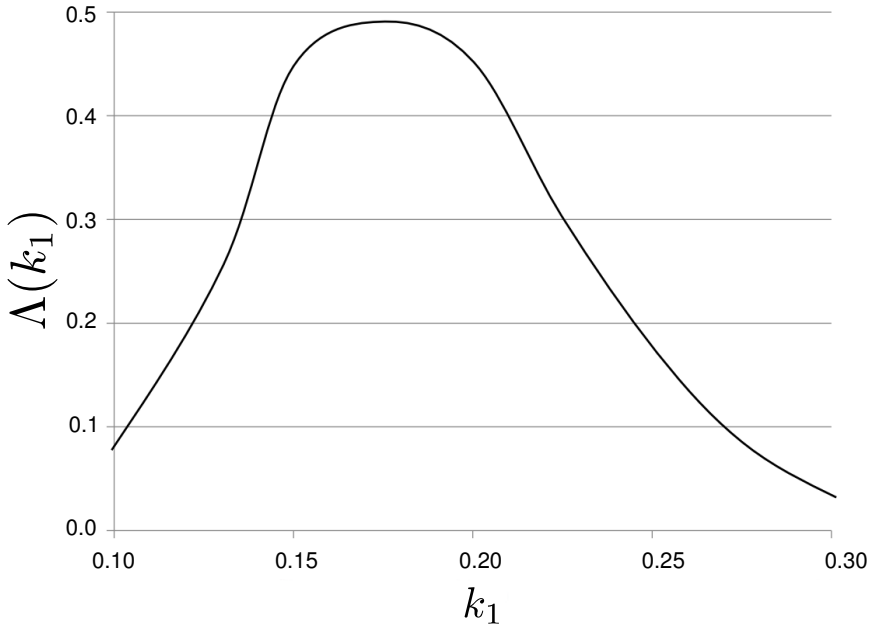
pCTMC + property



$$\phi = \mathrm{F}^{[1000,1000]}(X \geq 15 \wedge X \leq 20)$$

$$k_1 \in [0.1, 0.3] \quad k_2 = 0.01 \quad s_0 = [X]_0 = 15$$

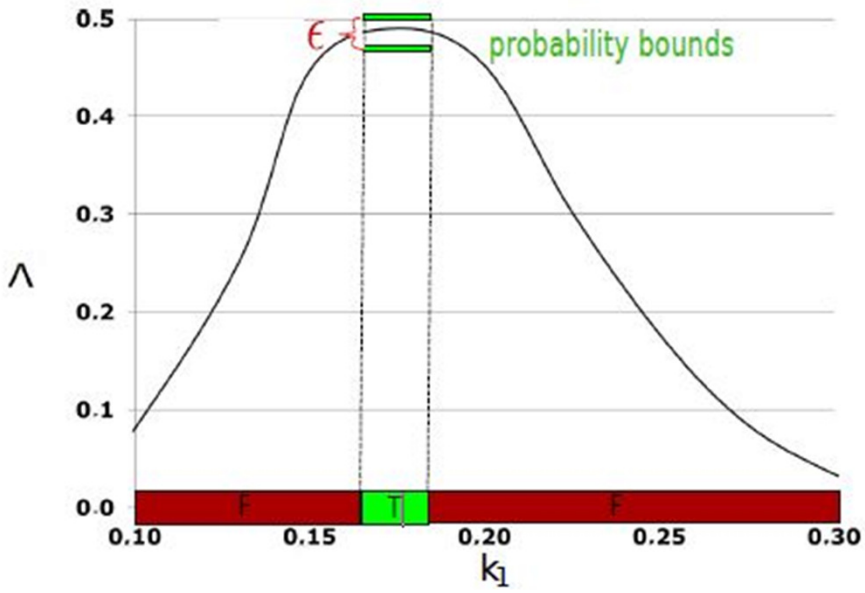
Satisfaction function



Max synthesis problem

For a given $\mathcal{P}$, ϕ and probability tolerance ϵ the problem is finding a partition $\{T, F\}$ of $\mathcal{P}$ and probability bounds $\Lambda^{\perp}$, $\Lambda^{\top}$ such that:

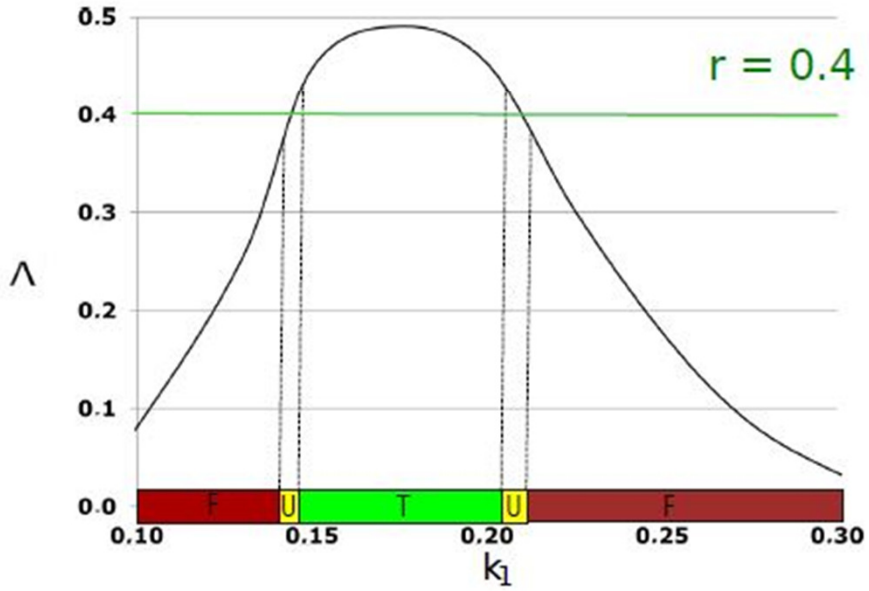
1. $\Lambda^{\perp} - \Lambda^{\top} \leq \epsilon$;
2. $\forall p \in T.\ \Lambda^{\perp} \leq \Lambda(p) \leq \Lambda^{\top}$; and
3. $\exists p \in T.\ \forall p' \in F.\ \Lambda(p) > \Lambda(p')$.



Threshold synthesis

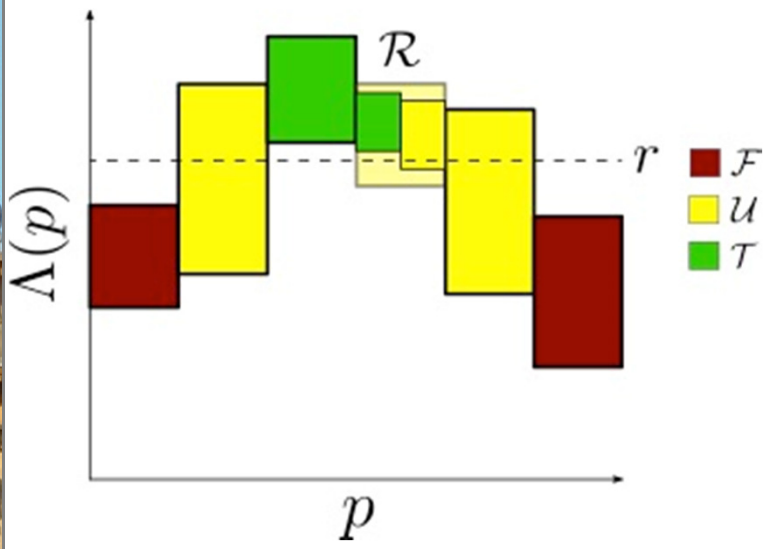
For a given $\mathcal{P}$, ϕ, probability threshold r and volume tolerance ε, the problem is finding a partition $\{T, U, F\}$ of $\mathcal{P}$ such that

1. $\forall p \in T.\ \Lambda(p) \geq r$; and
2. $\forall p \in F.\ \Lambda(p) < r$; and
3. $\mathrm{vol}(U)/\mathrm{vol}(\mathcal{P}) \leq \varepsilon$ ($\mathrm{vol}(A)$ is the volume of A).



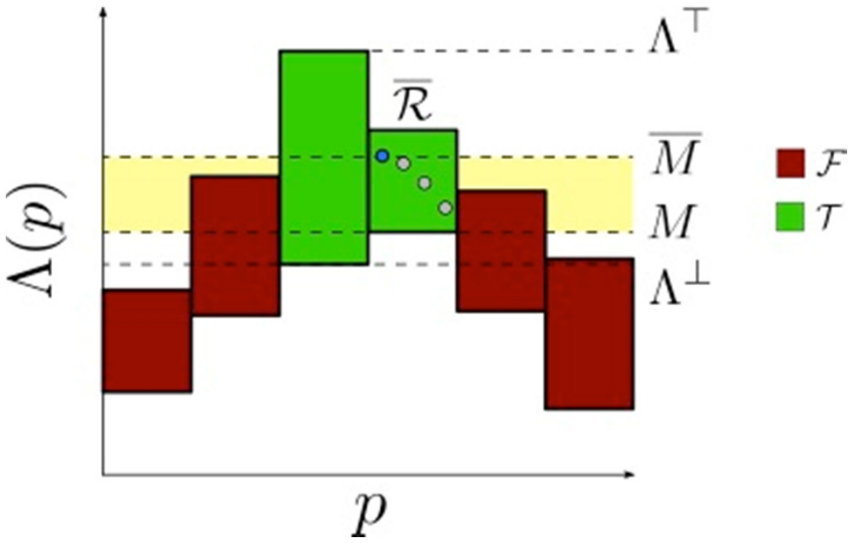
Example: synthesis

Threshold ($\geq r$)



- True if lower bound above r
- False if upper bound below r
- Undecided otherwise (to refine)

Max

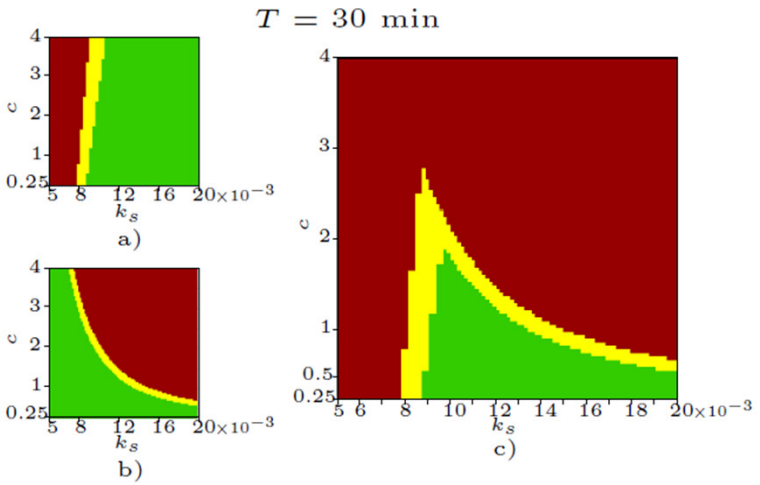


- False if upper bound below under-approximation of max prob M
- True otherwise (to refine)

DNA walkers: parameter synthesis

- Application to biosensor design: can we synthesise the values of rates to guarantee a specified reliability level?
- For the walker model:
  - walker stepping rate k = funct (k_s,c) where k_s lies in interval [0.005,0.020], c in [0.25, 4]
  - find regions of values of k_s and c where property is satisfied

a) $\Phi_1 = P_{\geq 0.4}[F^{[30,30]}$ finish-correct$]$
b) $\Phi_2 = P_{\leq 0.08}[F^{[30,30]}$ finish-incorrect$]$
c) $\Phi_1 \wedge \Phi_2$



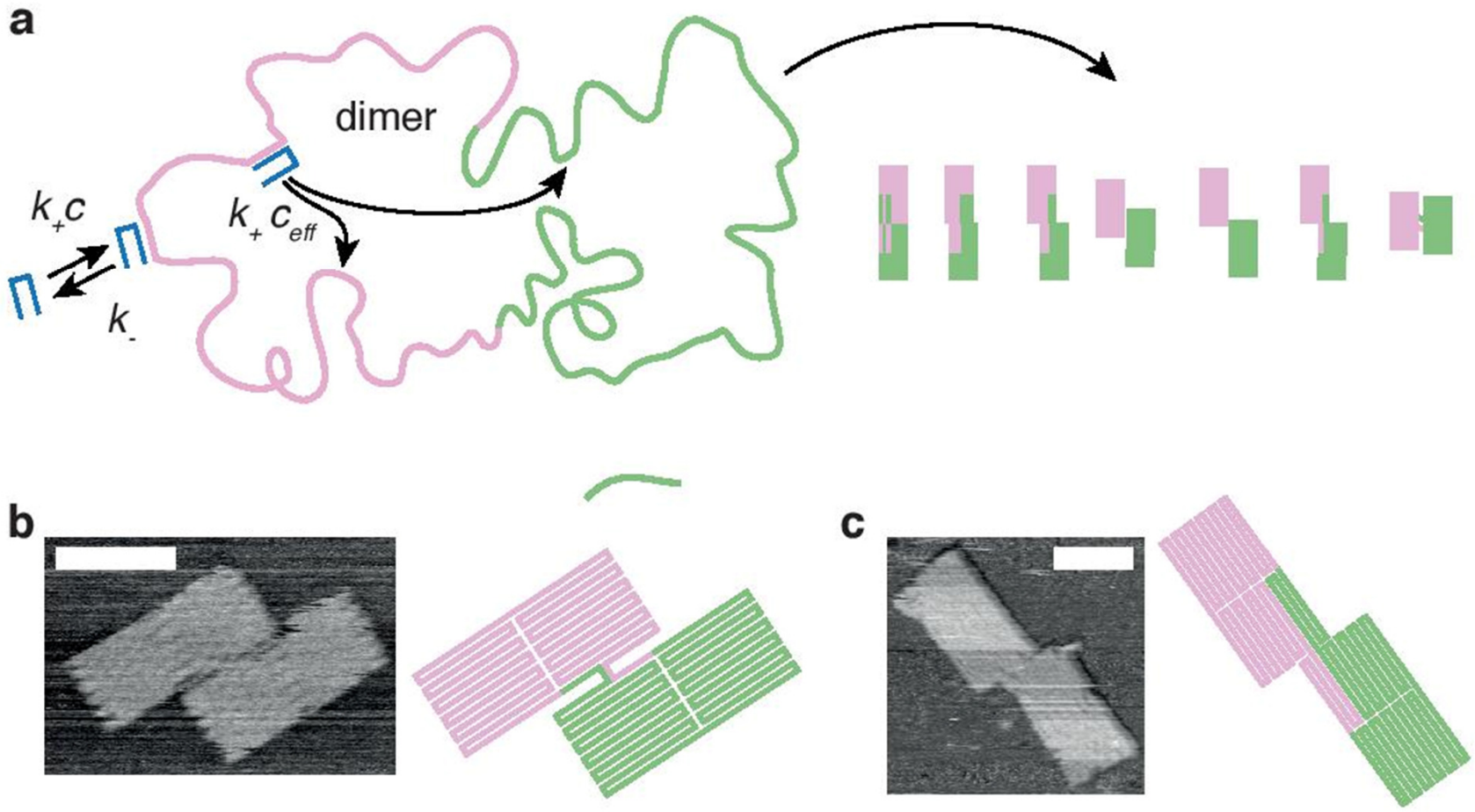
- Fast: for T=200, 88s with sampling, 329 subspaces

3. Modelling DNA origami

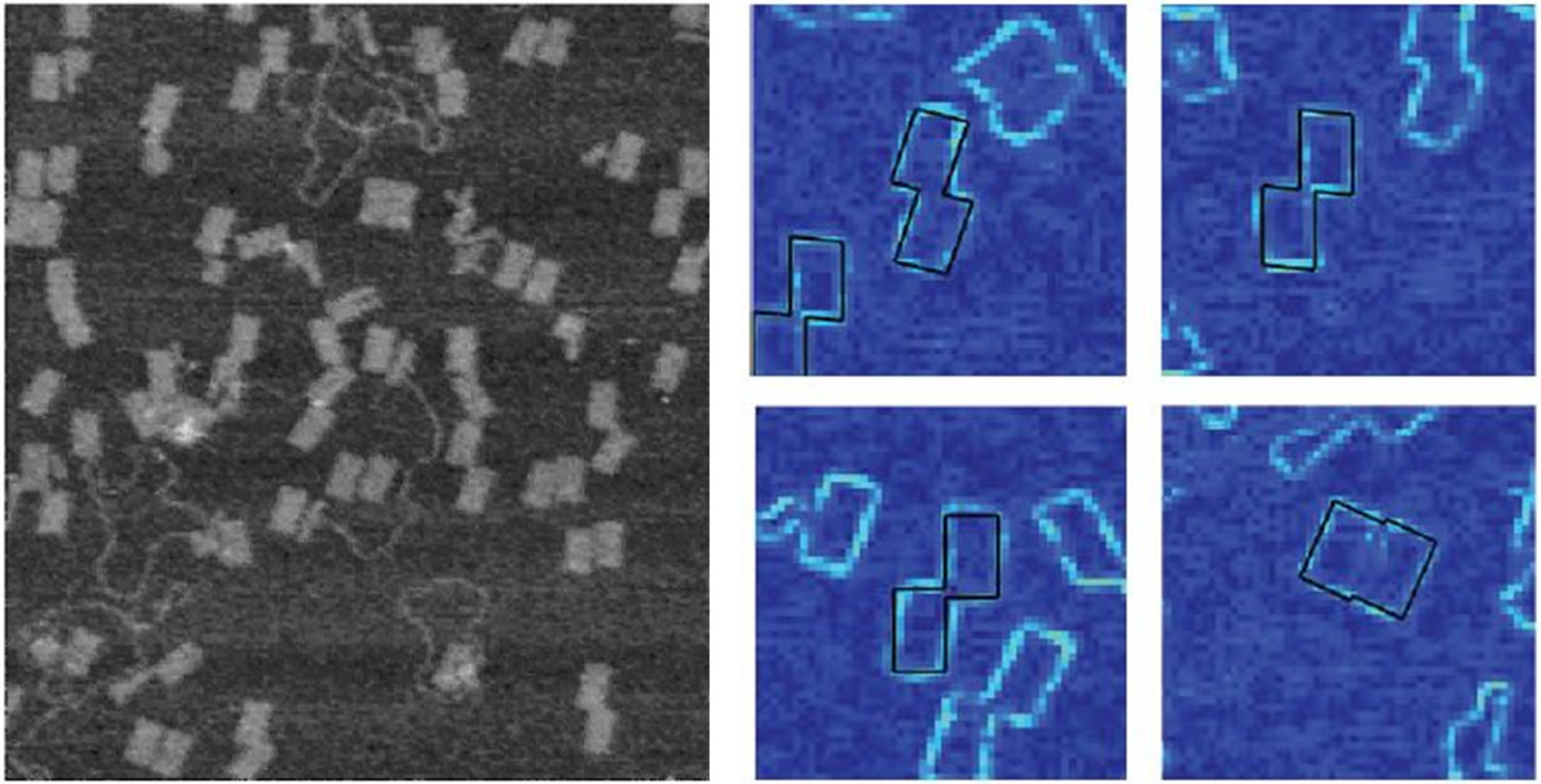
- DNA origami robust technique
  - robust assembly technique
  - monomer folds into the single most stable shape
- Aim to understand how to control the folding pathways
  - develop a 'dimer' origami design, which has several well-folded shapes (planar and unstrained) corresponding to energy minima
  - formulate an abstract CTMCmodel that is thermodynamically self-consistent
  - obtain model predictions using Gillespie simulation
  - perform a range of experiments (e.g. removing or cutting staples in half) that favour certain well-folded shapes
- Remarkably, the model is consistent with experimental observations

Guiding the folding pathway of DNA origami. Dunne, Dannenberg, Ouldridge, Kwiatkowska, Turberfield & Bath, Nature (in press)

Dimer origami



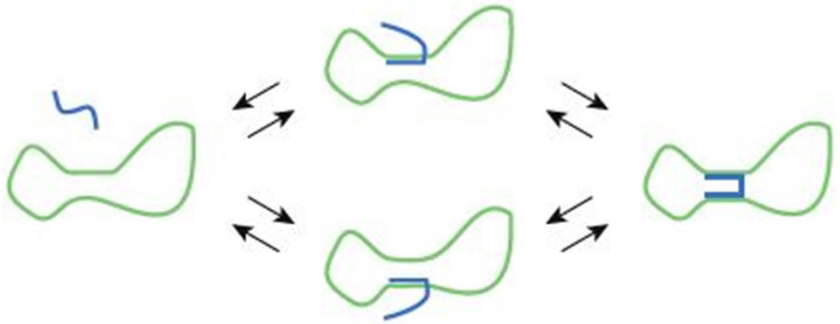
Dimer shapes



- Develop image processing software to classify shapes

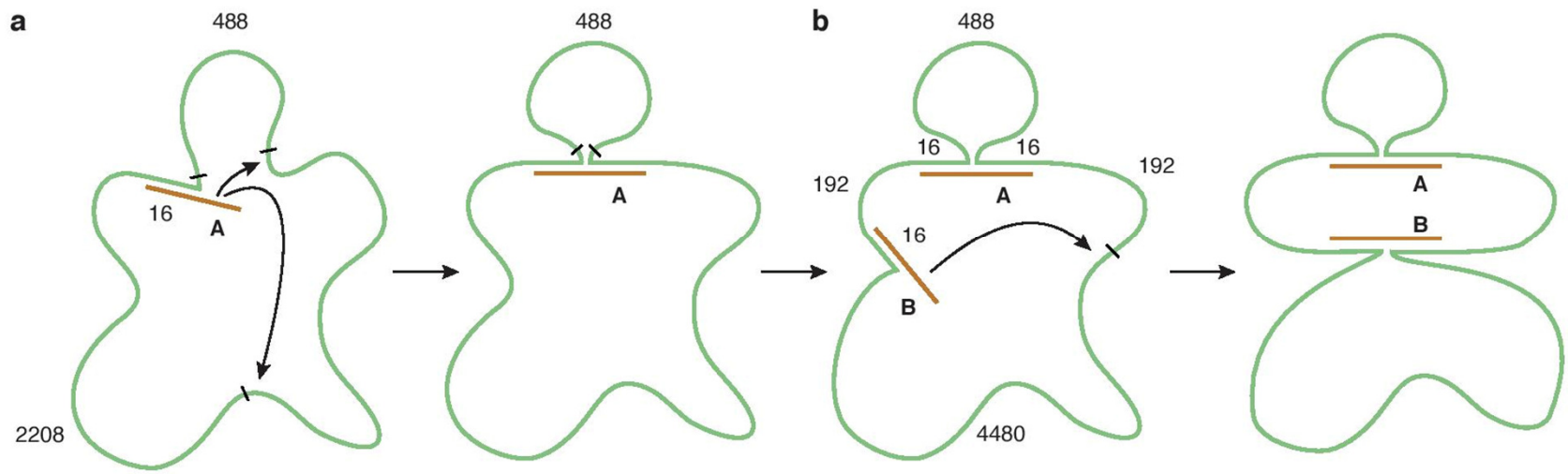
The CTMC model

- Abstract the scaffold as a sequence of domains (16nt)
  - each staple has 2 positions to bind to
  - single-domain and two-domain staples
- State space
  - for monomer, 5 possibilities for two-domain staples
  - for dimer, $4^N \times 34^M$,
    N = 24 one-domain and
    M = 156 two-domain staples



- Rates (inhomogeneous CTMC)
  - can use mass action only for staple binding from solution
  - otherwise, estimate free energy change
  - need to consider loop formation…

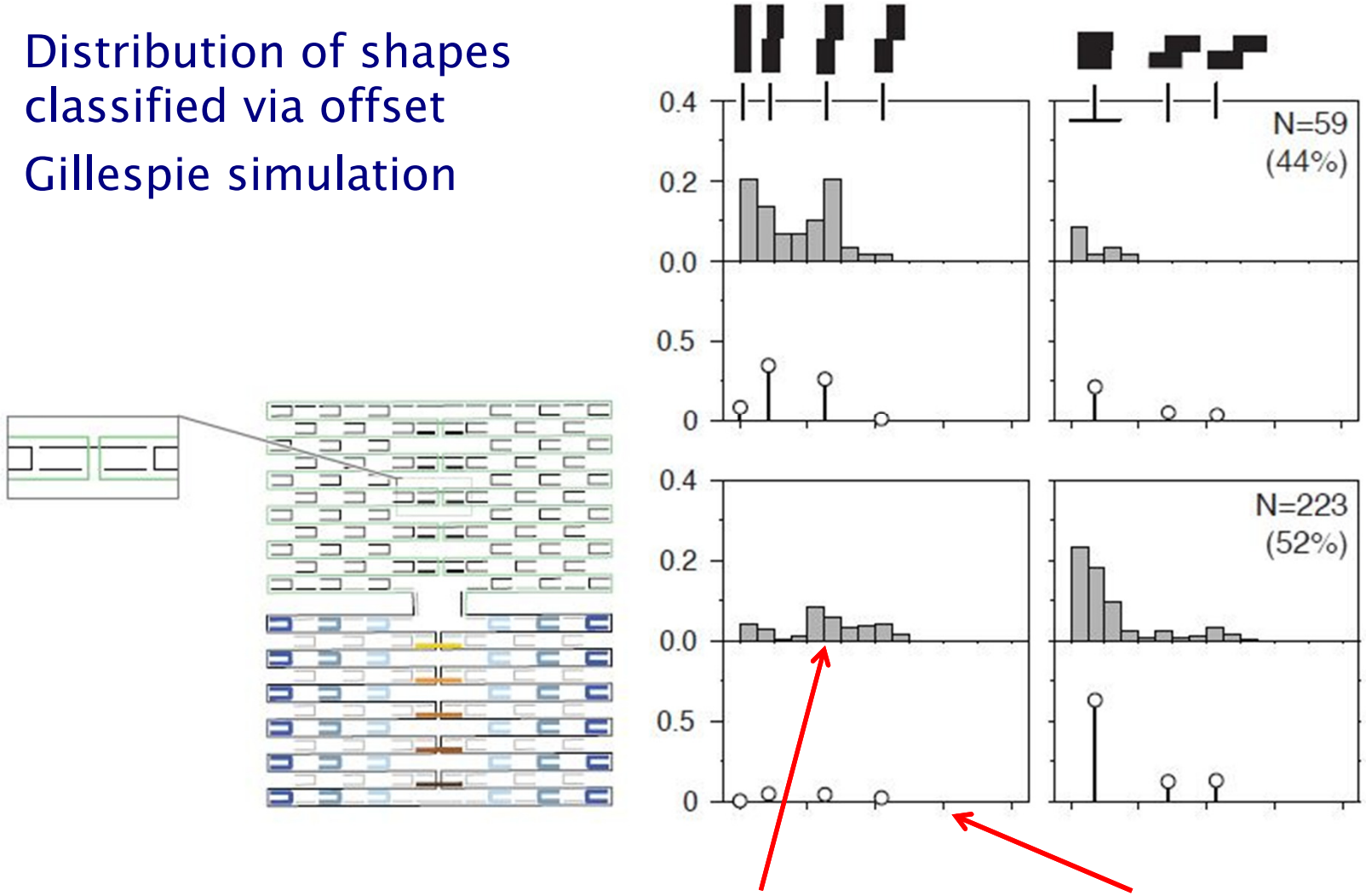
Loop formation



- Main idea: shortening of the loop by staple binding increases stability
  - use Dijkstra's shortest path algorithm to calculate adjustment in free energy
- Thus presence of staple A accelerates hybridization of B
- Planarity constraints

Results on folding

- Distribution of shapes classified via offset
- Gillespie simulation



Modified tile (broken/absent staples) Observed Predicted

What has been achieved?

- Established successfully
  - automatically found a flaw in DNA program
  - proposed design automation for DNA walker circuits, can guarantee reliability levels, fast
  - improved scientific understanding of DNA origami folding
- But limited scalability (but see [CMSB 2015])
  - DNA transducer: 6–7 molecules
  - DNA walker circuits: smaller models can be handled with fast adaptive unformisation, lager ones only with statistical model checking, sometimes with better accuracy
  - DNA origami folding: only simulation is feasible
- Challenges
  - need to incorporate physics (thermodynamics, entropy, energy), improve reliability

Conclusions

- Demonstrated that quantitative/probabilistic verification can play a central role in design automation of molecular devices

- Many positive results:
  - predictive models
  - successful experimental validation
  - demonstrated practical feasibility of probabilistic modelling and verification in some contexts

- Key challenge (as always): state space explosion
  - can we exploit compositionality in analysis?
  - can we synthesise walker circuit layout? origami designs?
  - parameter/model synthesis for more complex models…

Acknowledgements

- My group and collaborators n this work
- Project funding
  - ERC, EPSRC, Microsoft Research
  - Oxford Martin School, Institute for the Future of Computing

- See also
  - VERIWARE www.veriware.org
  - PRISM www.prismmodelchecker.org